PROTEXT

VERSION 5

User Guide and Reference Manual

ARNOR

PROTEXT

Version 5

WORD PROCESSOR

IBM PC XT/AT, PS/2 and Compatibles
Atari ST and TT
Commodore Amiga
Acorn Archimedes

Copyright (c) 1987-90 Arnor Ltd.
Issue 6, 1990 (for v5.04)

Protext program (c) 1987-1990 Arnor Ltd.
Amiga Workbench TM version 1.3 (c) 1985-1988 Commodore-Amiga Inc.
The Proximity/Collins Linguibase (c) 1985 William Collins Sons & Co. Ltd.
Legal and Medical Supplements (c) 1982 Merriam-Webster, Inc.
(c) Copyright 1982,1985 All Rights Reserved Proximity Technology Inc.

All rights reserved. It is illegal to reproduce or transmit either this manual or the accompanying computer program in any form without the written permission of the copyright holder. Copying of the supplied discs is permitted only to make security backup copies or to install onto a hard disc. The owner of this program is licensed to use one copy only at any given time. Network users require a special licence from Arnor. Software piracy is theft.

This manual was written using Protext and printed from camera-ready copy produced by Protext on a Kyocera F-2200S laser printer.

Program written by Arnor. Manual written by David Foster and Arnor.
Arnor Ltd., 611 Lincoln Road, Peterborough PE1 3HA, England. 0733 68909.

Acknowledgements

We would particularly like to thank David Foster for all his help, and all those who have contributed to the development of Protext. Protext is developed using the Zortech C compiler on the PC, the Lattice compiler on the ST and Amiga and the Acorn ANSI C compiler on the Archimedes.

Trademarks

Acorn and Archimedes are trademarks of Acorn Computers Ltd.
Amiga is a registered trademark of Commodore-Amiga Inc.
AmigaDOS is a trademark of Commodore-Amiga Inc.
Amstrad is a registered trademark of Amstrad plc.
Atari and ST are trademarks of Atari Corp.
Centronics is a registered trademark of Centronics Corp.
Collins is a trademark of William Collins Sons & Co. Ltd.
Commodore is a trademark of Commodore Electronics Ltd.
Diablo is a trademark of Xerox Corp.
Epson is a trademark of Epson America Inc.
First Word is a trademark of GST Holdings Ltd.
GEM, GEM Desktop and CP/M are trademarks of Digital Research Inc.
IBM, IBM PC, PC-DOS, and PS/2 are registered trademarks
of International Business Machines Corp.
Juki 6100 is a trademark of Tokyo Juki Industrial Co. Ltd.
Kyocera is a trademark of Kyocera Corp.
LaserJet is a trademark of Hewlett Packard Corp.
Merriam-Webster is a registered trademark of Merriam-Webster Inc.
MS-DOS is a registered trademark of Microsoft Corp.
Prodata is a trademark of Arnor Ltd.
Protext is a trademark of Arnor Ltd.
Proximity and Linguibase are registered trademarks of Proximity Technology Inc.
WordStar is a registered trademark of WordStar Corp.
Workbench is a trademark of Commodore-Amiga Inc.

This software may only be used by one person at a time. It must not be loaded or installed into more than one machine at any one location. If it is required on two machines then two copies must be purchased. It is, however, perfectly acceptable for the owner of the program to use it at home and at work, but not for someone else to use it at home while the owner is at work.

Please abide by these conditions and do not give copies to anyone else. If you do this you are cheating the developers of the software out of their rightful income.

Contents

1. Introduction 13

What is 'Word processing'? 13
What is Protext? 13
Equipment requirements 14
How to start 15

2. Getting Started - IBM PC and Compatibles 16

The supplied master discs 16
Make backups of the Protext discs 16
Hard disc installation 17
Floppy disc installation 18
Printer installation 20
Mouse installation 21
Loading and running Protext 21
Passing filenames 22
Configuration of Protext 22

3. Getting Started - Atari ST 23

The supplied master discs 23
Make backups of the Protext discs 23
Hard disc installation 24
Floppy disc installation 25
Printer installation 26
Loading and running Protext 27
Running Protext from low resolution mode 28
Configuration of Protext 28
Access to Gem desktop 28

4. Getting Started - Amiga 29

The supplied master discs 29
Make backups of the Protext discs 29
Hard disc installation 30
Floppy disc installation 31
Printer installation 32
Loading and running Protext 33
Passing a filename and parameters to Protext 34
Configuration of Protext 34

5. Getting Started - Archimedes 36

The supplied master discs 36
Make backups of the Protext discs 36
Desktop and CLI 36
Hard disc installation 37
Floppy disc installation 38
Printer installation 38
Loading and running Protext 39
Configuration of Protext 40
Archimedes filenames and directories 40

6. Overview of Protext 41

About the manual 41
Version numbers, updates and README 42
How to proceed 42
Overview of Protext 43
Menu operation 43
Special keys 48
Conventions used 49
Sticky shift keys 49
Sections specific to one system only 50
Editing large files 50
Are large documents necessary? 52
Graphics and text modes 53

7. Entering and Correcting Text 54

On screen help 54
The status line 54
Entering text 55
Upper and lower case 55
Deleting 56
Inserting 57
Swapping characters 57
Undeleting all or part of a line 57
Insert and Overwrite mode 57
Moving the cursor more rapidly 58
Moving the cursor with the mouse 59
Moving to a specified page, line or column number 60
Place markers 60
Multiple markers 61
Special markers 61

Scrolling 61
Splitting and joining lines 62

8. The File Selector and Disc Utilities 63

Movement around the directory listing 63
Selecting a file or directory 64
Viewing in a different format 64
Viewing different files 65
Changing drives 65
Creating a new directory 65
Viewing files 66
Erasing, Renaming and Copying files 66
File Protection 67
Directory Tree 67

9. Saving Text and other Simple Commands 69

Saving a document 69
Loading a document 69
Printing a document 70
Clearing the text 70
Listing the files saved on disc 70
Counting the words 70
Finding text 71
Quitting Protext 71

10. Cut and Paste Editing 72

Defining a block 72
Defining a block with the mouse 73
Moving or copying a block 73
Deleting a block 73
Undeleting a block 73
Alternative keys for block commands 74
Cut and paste editing - Box mode 74
Defining a box 74
Moving or copying a box 74
Deleting a box 75

11. Rulers Tabs and Margins 76

Ruler lines 76
Tabs 78
Normal Tabs 78
Decimal (right justification) Tabs 79
Centre Tabs 79
Indent Tabs 80
Ruler editing with the mouse 80
Margins 80
Left margins and tabs - applications 81

12. Formatting 83

Automatic Formatting 83
Word Wrap 83
Right Justification 83
Manual Re-formatting 84
Edit Mode Formatting 84
Command Mode Formatting 84
Centring text 85
Formatting whilst printing 85
Soft spaces, returns and hyphens 85
Non-break spaces and hyphens 86

13. Find and Replace 87

Options available 87
Using FIND 89
Using REPLACE 89
Wildcards in replace strings 90
Find and replace within block 90
Find word at cursor 90
Examples 91

14. Languages and Special Symbols 92

Accents 92
Special characters 93
The choose character window 93
Changing the keyboard layout 94
Greek letters 94
Line drawing 95

15. Miscellaneous Editing Commands 96

Timed automatic save 96
Add column or row of figures 96
Date and time 96
Viewing the text without control codes 97
Viewing the text with tabs, hard returns and hard spaces 98
Page break lines 98
Auto indent 99
43 and 50 line displays 99
Graphics and text mode 99
Window resizing 100

16. Multiple File Editing 101

Copying a block between documents 102
Split screen editing 103
Linked files 103

17. Printing 104

Printer drivers 104
What are printer control codes? 105
How to use printer control codes 105
Character pitch 106
The Print dialogue box 107
Printing commands 107
What determines the form of the printed output? 109
Background printing 110
Microspacing 111
Printing in program mode 111
Newspaper column printing 112

18. Proportional Printing 113

Selecting styles and fonts 115
The default font 116
Selecting a font 117
How to print text to a given width 119

19. Spelling Checker 121

What can the Spelling Checker do? 121
How the spelling checker decides what is a word 121
What sort of errors will it find? 122
Dictionary Files 123
The quick dictionary 124
Spell Checking a Document 124
Where to keep the dictionary 124
Using the Spelling Checker 125
All or part of a document during editing 125
Single word checking 128
Checking an entire text file or a block 128
At the end of the checking operation 129
Large files 130
Marking words so they will not be checked 130
Spell check whilst typing 130
Spelling Checker Utilities 131
Menu operation 131
Utility Commands 131

20. Index and Contents 132

Indexing 132
How to generate the index 134
Contents generating 136

21. Macros and Exec Files 138

Macros 138
Macro record mode 138
Pre-defined tokens 139
Using macros 140
Defining macros using the MACRO command 140
Saving macros for regular use 140
Exec Files 141
What is an exec file? 141
Creating an exec file 141
Using exec files 143
Using an exec file when loading Protext 143

22. Command Mode 144

Command mode Help 144
Command entry 144
Abbreviations 145
The current filename 146
Directories 146
Search path and temporary text path 147
Filenames 147
Backup files 148
Changing discs and drives 149
Scroll lock 150
Wildcards 150
Running External Programs from Protext 152

23. Commands 154

The Commands 154

24. Stored Commands Overview 195

Template Files 196
Inter paragraph space 197
Tab leader 197
Headers, footers and page numbers 198
Pagination 200
Footnotes and Endnotes 202

25. Stored Commands 204

The Stored Commands 204
Key to Syntax 229
The Expression Evaluator 230

26. Mail Merging 234

What is Mail Merging? 234
Data Files 235
What is a data file? 235
Data file format 235
How Protext reads data 236
Examples of valid data file records. 237
Exporting data from other programs. 238

Using Prodata with Protext 240
Simple Mail Merging 240
Getting started 241
Creating a Data file 241
Reading the data 242
Comparing RV and RU 245
Creating data by other means 247
Inserting separate text files 248
Address labels 250
Conditional Printing and Mail Merging 251
Advanced Mail Merging 255
More new commands 256
Closing data files 256
Special conditional commands 257
Repetitive looping 258
Writing to a disc file during printing 260
Displaying messages during printing 263
More on the RV stored command 264
More on the IN stored command 265
Making the most of Variables 265
Joining variables and text 266
Splitting variables 269
Using variables in stored commands 270
Numeric arithmetic 271
Date and time 273
Other system variables 274
Self incrementing variables and paragraph numbering 274
Roman numerals 275

27. Configuration 276

Using a CFG environment variable 276
Using CONFIG 277
Configuration options 279
Command line options 287
System Configuration Options 288

28. File Sorting 289

Running FSORT from Protext command mode 289
Running FSORT as a stand alone program 289
Sorting options 289
Some examples of using FSORT 294

29. File Conversion 295

Using CONVERT from the menu 295
Using CONVERT from the command line 295
User Defined Conversions - CVT files 298

30. Creating Printer Drivers 300

(i) General commands 301
(ii) Accent definition commands 306
(iii) Font definition commands 307
Defining fonts 310
The default printing font 311
Portability and Compatibility with Protext version 4.3 311
Setting up a printer driver for microspacing and proportional printing 313
Using a new printer driver 315

31. Hints and Tips 316

Printer Problems 316
Keyboard Problems 320
General Hints 322

A1. Summary of Protext Commands 323

(a) Edit mode 323
Cursor movement 323
Insertion and deletion 324
Block commands 324
Formatting and rulers 325
Find and replace, marker 325
Other commands 325
Spell checking 326
Extra characters 326
Accents and special characters 327
Miscellany 327
(b) Stored commands 328
Paper layout commands 329
Page formatting commands 329
Miscellaneous commands 330
Printer control commands 330
Variable and data input - mail merging 331
Conditional printing and mail merging 331

(c) Command Mode 332
Document handling 332
Printing options 333
Printer control commands 333
Sundry printing options 334
Text manipulation and formatting 334
Spell checking 334
Macro and Exec commands 335
Miscellany 335
Disc management commands 336
(d) External utility program commands 336

A2. Protext - Compatibility between versions 337

A3. Alternative Keyboard Commands 338

Using the MS-DOS keyboard driver 338
WordStar emulation 339

A4. Sample Merging Documents 340

A5. Simple letter example 346

Creating a Template File 346
Writing a letter 348

A6. Using a RAM disc 350

(a) PC 350
(b) Atari ST 352
(c) Amiga 353

A7. Protext key tokens 355

A8. IBM PC Character Set 361

A9. ST Character Set 362

A10. Amiga and Archimedes Character Set 363

A11. Glossary 364

Index 370

1. Introduction

What is 'Word processing'?

In its simplest form word processing is using a computer as a very sophisticated typewriter. Instead of typing a letter straight onto paper, the letter is displayed on the screen as it is typed.

Every word processor then allows you to:

(a) correct or change any part of the text.
(b) store the text on disc for later use.
(c) send the text to a printer.

An ordinary typewriter effectively only allows (c) because text cannot be changed or stored.

The advantages of even this simple form of word processing are enormous:

- it allows easy correction of mistakes, either immediately or at any later time.
- a number of similar documents can be produced with minimum effort.
- any number of copies can be printed without retyping the document.
- a disc can be given to someone else to use or add to.

Word processing offers far more than this, and Protext provides many powerful features - yet all are designed to be easy to use.

What is Protext?

Protext is a fully integrated word processing system, comprising word processor, spelling checker, mail merge, and file conversion and sorting programs. It is designed as a package which provides a complete environment from which all aspects of word processing and file management can be carried out without the need to load individual programs. Once Protext is loaded, there is no need to leave it until the day's word processing is completed. Documents may be created and the spelling checked, and then printed without leaving Protext.

Protext has been under continuous development since 1984. Each version is specially written to take full advantage of the particular operating system under which it runs, unlike many programs which are designed to run on a variety of machines and as a result suffer from the inevitable compromises imposed by such a requirement.

A high degree of compatibility has been built into this version of Protext and files created on other versions of Protext can be loaded and edited with the minimum of effort. Conversion programs are provided to carry out any changes, if necessary. Similarly, documents created with this version may be used on other versions. This compatibility extends also to the controls and commands used, within the limitations imposed by the different keyboards and operating systems. Anyone used to other versions of Protext will instantly feel at home with it and will only have to learn any new features incorporated into this version.

Protext is designed in such a way that it is easy to learn. Once the basic principles have been mastered, with the aid of 'tutorial' files and the comprehensive manual, documents may be easily created. All the extra features that are a part of Protext can be learnt one at a time as they are needed, or when the user feels ready to learn them. Unlike many programs, there is no need to learn it all at once.

Equipment requirements

One of the following computer systems (note that the correct version of Protext must be used - this information is shown on the packaging and the discs):

(i) IBM PC, XT, AT, PS/2 or compatible computer.
PC-DOS or MS-DOS version 2.00 or higher.
512K RAM minimum (640K recommended).
Optionally, a mouse.

(ii) Atari ST or TT with at least 1Mb RAM.

(iii) Commodore Amiga with at least 1Mb RAM and Workbench 1.2 or later.

(iv) Acorn Archimedes.

Printer. Any printer can be used, though some symbols are not available on all printers.

Printing line graphics requires a printer which supports the IBM character set, or an Epson or LaserJet compatible printer.

Some of the accented characters (especially those used in Eastern European languages) are only supported by a limited range of printers, including:

1. Certain 24 pin Epson compatible printers, including Star LC24.
2. HP LaserJet (II, IID, 2000, III or later model) using a font cartridge with the Desktop symbol set.
3. Kyocera laser printers with firmware version 6.37 or later.

See the 'Protext printer drivers' booklet for details of the facilities provided by individual printers and information about additions to this list.

How to start

Now turn to the appropriate 'Getting Started' chapter for your computer:

IBM PC and compatibles	chapter 2
Atari ST	chapter 3
Commodore Amiga	chapter 4
Acorn Archimedes	chapter 5

After installing Protext as described in one of these sections, turn to chapter 6 which gives a general introduction to Protext and the use of this manual.

2. Getting Started - IBM PC and Compatibles

This chapter explains how to configure Protext for the correct disc drives and printer and how to start using Protext. This is an important chapter and should be read carefully.

The supplied master discs

Protext is supplied on three or more discs including the Program Disc, the Dictionary Disc and the Printer Driver Disc. The Program Disc may contain a file called 'README' which gives the information about the latest version of the program.

WARNING: All the programs, files and the dictionary provided on the master discs are subject to copyright laws and copies may be made of these files and the dictionary, for your own use on one machine only. It is an offence to give, hire or sell copies of copyrighted material to other parties.

Protext operates under MS-DOS and is suitable for use on all 'IBM Compatible' computers. It may be used with single drive computers, but will take full advantage of two or more drives.

Make backups of the Protext discs

Before Protext is used, backup copies of the supplied discs MUST be made. Make sure that the discs are 'write protected' before copying them. With 5¼" discs, this means that the square notch cut out of the right hand edge of the disc should be covered with a small label or piece of sticky tape. With 3½" discs, the 'write protect tab' should be moved so that the hole is exposed.

To copy a disc first put the DOS system disc (supplied with the computer) in drive A, have a blank disc ready and type:

```
        DISKCOPY  A:  B:
or      DISKCOPY  A:  A:   if drives A and B are different sizes
```

Messages on the screen will indicate when to insert the discs. Repeat this process for each of the supplied discs. Put the original discs away to be retained as a security backup.

Hard disc installation

The installation process will create a subdirectory called 'PROTEXT' into which all the necessary Protext files will be copied. A subdirectory of 'PROTEXT' will also be created, called 'TEXT'. This will be used for saving documents unless otherwise specified. Before installing Protext, it is important to ensure that there is sufficient free space available on the disc by using DIR. There should be at least 1M byte (1000K) free on the hard disc to allow room for Protext, the associated programs and dictionary, and working files.

Put the Program Disc in drive A or B and select this drive by typing:

```
        A:
or      B:
```

Then, if the hard disc is drive C, type:

```
        INSTALL  C
```

If the hard disc is any other letter type the appropriate letter instead of 'C' in the above line. All the files will be copied to the hard disc, then the configuration menu will appear.

Press I to select the 'Initial Installation' menu. This menu will show the settings for the hard disc letter and the keyboard type and language. These can be changed using ↓, ↑ and RETURN. At this point the required colours and printer drivers may also be set up, though this may be left until later. If the hard disc is not C, D, E, or F, it will be necessary to use other CONFIG options to set the path and temporary path.

Keyboard configuration

It is important to configure the keyboard correctly. The keyboard language should be set to 'UK English' for U.K. users. There are three possible keyboard types. If the keyboard has a cursor key cluster separate from the numeric keypad then the 'Extended AT' type should be selected. Otherwise select the 'Standard PC'. If it is subsequently found that the '\¦' and '#~' keys are the wrong way round, use CONFIG to change to the 'Alternate AT' layout.

Press ESC.
Press S and then RETURN to save the configuration.
Press Q to quit.

In the root directory on the hard disc there is a file called 'CONFIG.SYS' which, among other things, sets a value for the number of buffers to be used by MS-DOS when accessing discs. It is recommended that this value is set to at least 20 when using Protext to ensure fast operation. This can most easily be modified by typing the following immediately after leaving the configuration program:

```
BUF20
```

This will rename the original CONFIG.SYS to CONFIG.OLD and create a new one with BUFFERS = 20 added to the end of the original file.

To run Protext type:

```
PROTEXT
```

Floppy disc installation

(i) Dual drive machines

Put the working copy of the Program Disc in drive A and type:

```
CONFIG
```

Press I to select the 'Initial Installation' menu.
Press RETURN to change the disc configuration to 'Dual floppy'.
Follow the 'keyboard configuration' instructions, above.
Press ESC.
Press S and then RETURN to save the configuration.
Press Q to quit.

Type:

```
PROTEXT
```

(ii) Single drive machines

After copying the discs as described above, Protext can be used with no further installation. Simply put the Program Disc in drive A and type:

```
PROTEXT
```

(iii) Using a startup disc

This section is optional reading.

Although Protext may now be used it will be necessary to 'boot' from the DOS system disc first, then change to the Protext disc and type 'Protext'. It is more convenient to use a Protext 'startup' disc.

A 'startup' disc is one which, when created by following the instructions here, will contain all the necessary System files as well as the Protext files. It will enable the computer to automatically load and run Protext when the disc is inserted at the start of the day, without the need to insert different discs.

To create a 'startup' disc format a blank disc as system format (using the command 'FORMAT A: /S'.

There will not be room for all the files. The files that should be copied are:

COMMAND.COM (the FORMAT command will copy this)
PROTEXT.EXE
PROTEXT.CFG (this is added when the printer is installed as below)
AUTOEXEC.BAT
CONFIG.SYS
PROTEXT.BAT

plus the required printer driver file

Printer installation

Protext will not initially be configured for any particular printer. In this state it will print on any printer but not be able to use many of a printer's features. The CONFIG program is used to select a printer. This should be run from the DOS command prompt.

Floppy disc users should put the Program disc in drive A and type:

```
A:
CONFIG
```

Hard disc users just type:

```
CONFIG
```

Press I to select the 'Initial Installation' menu.
Press P to install printers.

Up to 5 printer ports may be present on the computer and it is possible to select a different printer for each. In most cases there will only be one printer so only one needs to be set.

First determine which is the default printer port. This is the port to which the most often used printer is connected. If the printer is connected to the standard parallel port (this is by far the most common) then leave this selection as it is ('PAR1'). If the printer is a serial printer, press RETURN until the selection changes to 'SER1'.

Next move the highlighted bar by pressing the 'cursor down' key until the line corresponding to the default port is reached. Then press RETURN. A list of printers will appear. Select one by moving the highlighted bar with the cursor keys to the required printer and then pressing RETURN.

If the required printer is not listed then it is likely that it is equivalent to one of those listed. In particular the 'standard dot matrix printer' will work for the majority of dot matrix printers ('Epson compatible' printers) and the 'standard proportional daisywheel' for daisywheel printers ('Diablo compatible').

After selecting the printer or printers press ESC twice to return to the main menu.
Press S and then RETURN to save the configuration. Press Q to quit.

Mouse installation

If a mouse is fitted, it can be used in Protext for moving the cursor. The mouse driver must be installed as described in the mouse documentation. This will probably entail either adding a line to AUTOEXEC.BAT or to CONFIG.SYS. If the mouse driver is correctly installed a mouse cursor will appear on the screen as a steady (not flashing) block cursor. Moving the mouse will move the mouse cursor on the screen. To move the editing cursor to this position press the left mouse button.

Loading and running Protext

Once the working copies of the disc have been created, or in the case of a hard disc, the files copied across, Protext may be loaded and run in the following way:-

Floppy Disc: From switch on, insert the DOS system disc to load MS-DOS. When MS-DOS has been loaded, insert the Program disc into drive A, select drive A as the current drive, if necessary, and type:-

PROTEXT and press RETURN.

When the program has loaded the program disc should be removed, and the text disc inserted in drive A.

On a two drive computer the Dictionary disc should be put in drive B. After loading the program the Program disc will be removed and the text disc put in drive A.

On a single drive computer the text disc will be put in drive A after the program has loaded. The Dictionary disc should be put into drive A when required for spell checking or accessing the help information. Alternatively the dictionary files may be copied onto the text disc, if room.

Important note: Documents written using Protext should be saved onto both the text disc and a backup text disc, for security. How to save text is explained later.

Hard Disc: It is only necessary to type Protext, as described above and Protext will load. If the normal default setting of PATH includes the root directory, Protext may be called from any directory and will load with that directory as the current directory.

Passing filenames

It is also possible to specify one or two files to be loaded at the same time as Protext is loaded and this is done by specifying the names after the 'PROTEXT' command. Drive paths may be specified.

For example:-

```
PROTEXT LETTER\MYLETTER
```

will load Protext and a text file called 'MYLETTER', which is in a subdirectory, called 'LETTER'.

```
PROTEXT MAIN.C MAIN.H
```

will load MAIN.C as the first document and MAIN.H as the second.

Configuration of Protext

Whilst Protext may now be used and is configured for the correct printer and disc drives, certain other parameters which are set to default values by the installation program may be changed at any time. Details of how to reconfigure Protext using CONFIG are given in the chapter 'Configuration'. It is recommended that some experience is gained with using Protext before any changes are made.

3. Getting Started - Atari ST

This chapter explains how to configure Protext for the correct disc drives and printer and how to start using Protext. This is an important chapter and should be read carefully.

Important note for 520ST users

Protext is normally supplied on double sided discs. Older 520STs cannot use these discs as they have a single sided disc drive. If the discs do not work and a 520ST is being used the discs should be returned to Arnor to be exchanged for single sided discs. Note that a memory upgrade to 1Mb is also required.

The supplied master discs

Protext is supplied on three or more discs including the Program Disc, the Dictionary Disc and the Printer Driver Disc. The Program Disc may contain a file called 'README' which gives the information about the latest version of the program.

WARNING: All the programs, files and the dictionary provided on the master discs are subject to copyright laws and copies may be made of these files and the dictionary, for your own use on one machine only. It is an offence to give, hire or sell copies of copyrighted material to other parties.

Protext operates on all Atari ST and TT machines with at least 1Mb of memory. It may be used with single drive computers, but will take full advantage of two or more drives. Use of a hard disc or a RAM disc will enhance the performance of Protext.

Make backups of the Protext discs

Before Protext is used, backup copies of the supplied discs MUST be made. Make sure that the discs are 'write protected' before copying them. The 'write protect tab' should be moved so that the hole is exposed.

The supplied discs should be copied from the GEM desktop by dragging the floppy disk A icon onto the floppy disk B icon and following the on-screen instructions. Put the original discs away to be retained as a security backup.

Hard disc installation

Installation of Protext onto a hard disc is extremely simple. Before installing Protext, it is important to ensure that there is sufficient free space available on the disc by clicking on the disc icon and using File Info. There should be at least 1M byte (1000K) free on the hard disc to allow room for Protext, the associated programs and dictionary, and working files.

First, create a new folder on the hard disc called Protext.
Second, copy all the files from the supplied discs into this new folder.

Double click on CONFIG.PRG (in the Protext folder). The configuration menu will then appear.

Press I to select the 'Initial Installation' menu. This menu will show the settings for the hard disc letter and the keyboard language. These can be changed using ↓, ↑ and RETURN. At this point the required colours and printer drivers may also be set up, though this may be left until later. If the hard disc is not C, D, E, or F, it will be necessary to use other CONFIG options to set the path and temporary path.

Press ESC.
Press S and then RETURN to save the configuration.
Press Q to quit.

Floppy disc installation

You should copy any required desktop accessories onto the working copy of the program disc. In particular the file CONTROL.ACC will probably be needed. Finally you can save your preferred desktop configuration on the same disc. This is done by using the 'save desktop' option in the options menu. This will create a file called DESKTOP.INF. This disc should now be labelled 'Protext Startup disc'.

(i) Dual drive machines

Put the Startup Disc in drive A and double click on CONFIG.PRG

Press I to select the 'Initial Installation' menu.
Press RETURN to change the disc configuration to 'Dual floppy'.
Press ESC.

Press S and then RETURN to save the configuration.
Press Q to quit.

Double click on PROTEXT.PRG

(ii) Single drive machines

After copying the discs as described above, Protext can be used with no further installation. Simply put the Program Disc in drive A and double click on PROTEXT.PRG.

Printer installation

Protext will not initially be configured for any particular printer. In this state it will print on any printer but not be able to use many of a printer's features. The CONFIG program is used to select a printer. This should be run from the GEM desktop.

Floppy disc users should first put the Startup disc in drive A.

Double click on CONFIG.PRG.

Press I to select the 'Initial Installation' menu.
Press P to install printers.
It is possible to select a different printer for the serial and parallel ports, if both are present. In most cases there will only be one printer so only one needs to be set.

First determine which is the default printer port. This is the port to which the most often used printer is connected. If the printer is connected to the standard parallel port (this is by far the most common) then leave this selection as it is ('PAR1'). If the printer is a serial printer, press RETURN until the selection changes to 'SER1'.

Next move the highlighted bar by pressing the 'cursor down' key until the line corresponding to the default port is reached. Then press RETURN. A list of printers will appear. Select one by moving the highlighted bar with the cursor keys to the required printer and then pressing RETURN.

If the required printer is not listed then it is likely that it is equivalent to one of those listed. In particular the 'standard dot matrix printer' will work for the majority of dot matrix printers ('Epson compatible' printers) and the 'standard proportional daisywheel' for daisywheel printers ('Diablo compatible').

After selecting the printer or printers press ESC twice to return to the main menu.
Press S and then RETURN to save the configuration.
Press Q to quit.

Atari laser printer

The Atari SLM804 laser printer works in a different way from other printers, in that it is controlled entirely by resident software in the computer. This software comes in the form of desktop accessories and is (or will be) available to enable the printer to emulate various types of printer. To use the laser printer with Protext you must use an emulation matching one of Protext's printer drivers.

There are three main printer types that are commonly provided as emulations in laser printers. These are 'Epson', 'Diablo' and 'Laserjet'. Each of these offers different facilities and Protext includes a printer driver for each.

A Diablo emulation is supplied with the Atari laser printer. An Epson emulation is also available. There is no news of a Laserjet emulation yet. To use the Diablo emulation, the desktop accessory must be loaded into the desktop (consult the laser printer documentation for details) and Protext must be configured to use the SLM804D printer driver (see above). The Epson emulation is recommended and should be used with the SLM804E printer driver.

Loading and running Protext

Once the working copies of the disc have been created, or in the case of a hard disc, the files copied across, Protext may be loaded and run in the following way:-

Floppy Disc: From switch on, insert the Protext Startup disc and double click on PROTEXT.PRG. When the program has loaded the program disc should be removed, and the text disc inserted in drive A.

On a two drive computer the Dictionary disc should be put in drive B. After loading the program the Program disc will be removed and the text disc put in drive A.

On a single drive computer the text disc will be put in drive A after the program has loaded. The Dictionary disc should be put into drive A when required for spell checking or accessing the help information. Alternatively the dictionary files may be copied onto the text disc, if room.

Hard Disc: Double click on the PROTEXT directory, then double click on PROTEXT.PRG.

Important note: Documents written using Protext should be saved onto both the text disc and a backup text disc, for security. Saving text is explained later.

Running Protext from low resolution mode

If the Gem desktop is using low resolution (40 column) mode, Protext will run in this mode. To run with an 80 column display on a colour monitor the 'Set Preferences' option in the Options menu should be used to select medium or high resolution mode before running Protext.

Configuration of Protext

Whilst Protext may now be used and is configured for the correct printer and disc drives, certain other parameters which are set to default values by the installation program may be changed at any time. Details of how to reconfigure Protext using CONFIG are given in the chapter 'Configuration'. It is recommended that some experience is gained with using Protext before any changes are made.

One change that you might want to make immediately is to choose to use the Gem file selector (or a replacement that you have installed) in Protext.

Access to Gem desktop

The left mouse button provides access to Gem desktop accessories from within Protext. When the mouse pointer is moved to the top line of the screen and the left mouse button is pressed the Gem menu bar will appear at the top of the screen. The mouse may then be used to call up any accessories in the usual way. To return to Protext the 'quit' option on the 'desk' menu should be selected.

4. Getting Started - Amiga

This chapter explains how to configure Protext for the correct disc drives and printer and how to start using Protext. This is an important chapter and should be read carefully.

The supplied master discs

Protext is supplied on three or more discs including the Program Disc, the Dictionary Disc and the Printer Driver Disc. The Program Disc may contain a file called 'README' which gives the information about the latest version of the program.

WARNING: All the programs, files and the dictionary provided on the master discs are subject to copyright laws and copies may be made of these files and the dictionary, for your own use on one machine only. It is an offence to give, hire or sell copies of copyrighted material to other parties.

Protext operates on all Commodore Amiga machines with at least 1Mb memory. Protext may be used with single drive Amigas, but will take full advantage of extra drives. A hard disc or RAM disc will enhance the performance of Protext.

Make backups of the Protext discs

Before Protext is used, backup copies of the supplied discs MUST be made. Make sure that the discs are 'write protected' before copying them. The 'write protect tab' should be moved so that the hole is exposed.

With the Workbench disc in drive df0:, select the Workbench disc icon, then choose 'Duplicate' from the 'Workbench' menu. Insert the supplied disc in drive df0: when the requester prompts for the '(FROM disc)' and replace it with the blank disc when asked to insert the disc to receive copy, until the disc has been copied. If a second drive is installed, the supplied disc should be inserted into drive df0: and the blank disc into df1: and there will be no further requests to insert discs.

Alternatively, use DISCCOPY from the CLI.

Hard disc installation

Installation of Protext onto a hard disc is extremely simple. Before installing Protext, it is important to ensure that there is sufficient free space available on the disc. There should be at least 1M byte (1000K) free on the hard disc to allow room for Protext, the associated programs and dictionary, and working files.

First, start the Amiga in the usual way so that the hard disc is active.

Workbench 1.2 users:	Double click on the CLI icon. If this is not present, run Preferences first and select the option to enable use of the CLI.
Workbench 1.3 / 2.0:	Double click on the SHELL icon which appears in the main workbench window.

Put the Protext disc 1 (Program disc) in drive df0:

Then, to install Protext (or update to a new version) type:

```
CD DF0:
EXECUTE HDINSTALL dh0:
```

(To install on a different hard disc partition replace 'dh0:' by the appropriate name, e.g. EXECUTE HDINSTALL jh1:)

When installing Protext for the first time the configuration menu will then appear.

Press I to select the 'Initial Installation' menu. This menu will show the settings for the hard disc letter and the keyboard language. These can be changed using ↓, ↑ and RETURN. At this point the required colours and printer drivers may also be set up, though this may be left until later. If the hard disc is not DH0:, DH1:, DH2:, or DH3:, it will be necessary to use other CONFIG options to set the path and temporary path. Note that the 'disc configuration' option has exactly the same effect as setting the search path and temporary path in the 'file and disc options' menu.

Press ESC.
Press S and then RETURN to save the configuration.
Press Q to quit.

After installing Protext the icons can be positioned as required. To save the new positions use the 'snapshot' option from the Workbench (see Amiga manual).

Floppy disc installation

The working copy of the program disc should be labelled 'Protext Startup disc'. The computer should be started up by putting in the Workbench disc, not the Protext disc.

(i) Dual drive machines - A500, A2000 with two internal drives

Put the Startup Disc in drive df0: and double click on the CONFIG icon.

Press I to select the 'Initial Installation' menu.
Press → to change the disc configuration to 'Dual floppy DF0:/DF1:'.
Press ESC.
Press S and then RETURN to save the configuration.
Press Q to quit.

Double click on the Protext icon.

(ii) A2000 with one internal, one external drive

Put the Startup Disc in drive df0: and double click on the CONFIG icon.

Press I to select the 'Initial Installation' menu.
Press → to change the disc configuration to 'Dual floppy DF0:/DF2:'.
Press ESC.
Press S and then RETURN to save the configuration.
Press Q to quit.

Double click on the Protext icon.

(iii) Single drive machines

After copying the discs as described above, Protext can be used with no further installation. Simply put the Program Disc in drive df0: and press CTRL-Amiga-Amiga. Alternatively, double click on the Protext icon from the workbench.

Printer installation

Protext does not make use of the Amiga printer drivers normally selected using Preferences. Instead, special printer drivers are supplied, covering a wider range of printers. The Protext printer drivers hold different information to the Amiga printer drivers and so Protext does not use the Amiga printer drivers at all. However, Preferences should still be used to choose either the parallel or serial port for printer output.

Protext will not initially be configured for any particular printer. In this state it will print on any printer but not be able to use many of a printer's features. The CONFIG program is used to select a printer. This can be run from the Workbench or CLI.

Floppy disc users should first put the Startup disc in drive A.

Double click on the CONFIG icon.

Press I to select the 'Initial Installation' menu.
Press P to install printers.

The currently selected printer driver will be shown (if one has been selected previously). Press RETURN. A list of printers will appear. Select one by moving the highlighted bar with the cursor keys to the required printer and then pressing RETURN.

If the required printer is not listed then it is likely that it is equivalent to one of those listed. In particular the 'standard dot matrix printer' will work for the majority of dot matrix printers ('Epson compatible' printers) and the 'standard proportional daisywheel' for daisywheel printers ('Diablo compatible').

After selecting the printer or printers press ESC twice to return to the main menu.
Press S and then RETURN to save the configuration.
Press Q to quit.

Loading and running Protext

Once the working copies of the disc have been created, or in the case of a hard disc, the files copied across, Protext may be loaded and run in the following way:-

Floppy Disc: From switch on, insert the Protext program disc and Protext will load automatically. When the program has loaded the program disc should be removed, and the text disc inserted in drive df0:.

On a two drive computer the Dictionary disc should be put in drive df1:. After loading the program the Program disc will be removed and the text disc put in drive df0:.

On a single drive computer the text disc will be put in drive df0: after the program has loaded. The Dictionary disc should be put into drive df0: when required for spell checking or accessing the help information. Alternatively the dictionary files may be copied onto the text disc, if there is sufficient space.

Hard Disc: Double click on the Protext drawer, then double click on the Protext icon.

Important note: Documents written using Protext should be saved onto both the text disc and a backup text disc, for security. Saving text is explained later.

Passing a filename and parameters to Protext

It is also possible to specify one or two files to be loaded at the same time as Protext is loaded and this is done by specifying the names after the 'Protext' command. Drive paths may be specified.

For example:-

```
PROTEXT MAIN.C MAIN.H
```

will load MAIN.C as the first document and MAIN.H as the second.

Command line options may be given by typing a hyphen ('-') followed by an option letter. Details are given in the chapter 'Configuration'. Alternatively the options may be typed into a file called PROTEXT.OPT (save this file in the root directory). This allows options to be used when running Protext from the Workbench. The most useful option allows the window size and position to be set (see below).

Configuration of Protext

Whilst Protext may now be used and is configured for the correct printer and disc drives, certain other parameters which are set to default values by the installation program may be changed at any time. Details of how to reconfigure Protext using CONFIG are given later in the manual. It is recommended that some experience is gained with using Protext before changes are made, other than those provided in the installation menu.

Keyboard layout configuration

One change that you might want to make immediately is to choose to use the keyboard type. The 'standard Amiga' layout uses the Intuition standard for the cursor movement and this may be preferred by those using other Amiga software packages that use the same keys. The 'standard Protext' layout uses the same keys as all other versions of Protext including v4.1 and v4.2 on the Amiga and versions for other computers. This manual describes the standard Protext keys throughout; the Amiga standard commands that differ from these are listed here with the standard Protext equivalents in brackets:

	SHIFT	CTRL	ALT
↑	Top of text (CTRL-[)	Symbol/line draw (ALT-↑)	Scroll page up (CTRL-↑)
↓	End of text (CTRL-])	Symbol/line draw (ALT-↓)	Scroll page down (CTRL-↓)
←	Start of line (CTRL-←)	Symbol/line draw (ALT-←)	Word left (SHIFT-←)
→	End of line (CTRL-→)	Symbol/line draw (ALT-→)	Word right (SHIFT-→)

Additionally SHIFT-CTRL with ↑ and ↓ perform the scrolling up or down one line functions (SHIFT with ↑ and ↓ in Protext keyboard layout).

Other installation options

An option in the installation allow the colours to be chosen for the different parts of the screen display. An option to use the 'ARP' file requester is provided. This is for compatibility with other programs that use the ARP file requester and 'arp.library' must have been previously installed.

Another option allows you to choose whether Protext creates icons for data files. Distinct icons are used for documents and program files. Clicking on an icon from Workbench will run Protext and automatically load the file. Multiple selection of icons is also supported. The icons themselves are defined in the files _doc.info and _prog.info which must be present in the search path directory. If _prog.info is deleted icons will be created for documents only.

5. Getting Started - Archimedes

This chapter explains how to configure Protext for the correct disc drives and printer and how to start using Protext. This is an important chapter and should be read carefully.

The supplied master discs

Protext is supplied on three or more discs including the Program Disc, the Dictionary Disc and the Printer Driver Disc. The Program Disc may contain a file called 'README' which gives the information about the latest version of the program.

WARNING: All the programs, files and the dictionary provided on the master discs are subject to copyright laws and copies may be made of these files and the dictionary, for your own use on one machine only. It is an offence to give, hire or sell copies of copyrighted material to other parties.

Protext operates on all Acorn Archimedes computers. It may be used with single drive computers, but will take full advantage of two or more drives. Use of a hard disc or a RAM disc will enhance the performance of Protext.

Make backups of the Protext discs

Before Protext is used, backup copies of the supplied discs MUST be made. Make sure that the discs are 'write protected' before copying them. The 'write protect tab' should be moved so that the hole is exposed.

The supplied discs should be copied as described in the Archimedes User Guide. Put the original discs away to be retained as a security backup.

Desktop and CLI

Protext can be run both from the Desktop and from the Command Line. When run from the desktop, Protext is 'RISC OS Compatible'. It does not use the RISC OS style menus but after quitting Protext the desktop configuration will be restored.

Hard disc installation

Before installing Protext, it is important to ensure that there is sufficient free space available on the disc. There should be at least 1M byte (1000K) free on the hard disc to allow room for Protext, the associated programs and dictionary, and working files.

First, create a new directory on the hard disc called Protext.

Second, copy all the files from the supplied discs into this new directory. The files contained in subdirectories on the supplied discs should be copied into the same subdirectories on the hard disc.

Double click on CONFIG (in the Protext directory). The configuration menu will then appear.

Press I to select the 'Initial Installation' menu. This menu will show the settings for the hard disc letter and the keyboard language. These can be changed using ↓, ↑ and RETURN. At this point the required colours and printer drivers may also be set up, though this may be left until later. If the hard disc is not 4, 5, 6 or 7, it will be necessary to use other CONFIG options to set the path and temporary path.

Press ESC.
Press S and then RETURN to save the configuration.
Press Q to quit.

Floppy disc installation

The working copy of the program disc should be labelled 'Protext Startup disc'.

(i) Dual drive machines

Put the Startup Disc in drive 0 and double click on CONFIG

Press I to select the 'Initial Installation' menu.
Press RETURN to change the disc configuration to 'Dual floppy'.
Press ESC.

Press S and then RETURN to save the configuration.
Press Q to quit.

Double click on PROTEXT

(ii) Single drive machines

After copying the discs as described above, Protext can be used with no further installation. Simply put the Program Disc in drive 0 and double click on PROTEXT.

Printer installation

Protext will not initially be configured for any particular printer. In this state it will print on any printer but not be able to use many of a printer's features. The CONFIG program is used to select a printer. This should be run from the desktop.

Floppy disc users should first put the Startup disc in drive 0.
Double click on CONFIG.

Press I to select the 'Initial Installation' menu.
Press P to install printers.

A list of printers will appear. Select one by moving the highlighted bar with the cursor keys to the required printer and then pressing RETURN.

If the required printer is not listed then it is likely that it is equivalent to one of those listed. In particular the 'standard dot matrix printer' will work for the majority of dot matrix printers ('Epson compatible' printers) and the 'standard proportional daisywheel' for daisywheel printers ('Diablo compatible').

After selecting the printer or printers press ESC twice to return to the main menu.
Press S and then RETURN to save the configuration. Press Q to quit.

Loading and running Protext

Once the working copies of the disc have been created, or in the case of a hard disc, the files copied across, Protext may be loaded and run either from the Desktop (by clicking on PROTEXT) or the Command Line (by typing 'protext').

Floppy Disc: From switch on, insert the Protext Startup disc and double click on PROTEXT. When the program has loaded the program disc should be removed, and the text disc inserted in drive 0.

On a two drive computer the Dictionary disc should be put in drive 1. After loading the program the Program disc will be removed and the text disc put in drive 0.

On a single drive computer the text disc will be put in drive 0 after the program has loaded. The Dictionary disc should be put into drive 0 when required for spell checking or accessing the help information. Alternatively the dictionary files may be copied onto the text disc, if room.

Hard Disc: Double click on the PROTEXT directory, then double click on PROTEXT.

Important note: Documents written using Protext should be saved onto both the text disc and a backup text disc, for security. Saving text is explained later.

Configuration of Protext

Whilst Protext may now be used and is configured for the correct printer and disc drives, certain other parameters which are set to default values by the installation program may be changed at any time. Details of how to reconfigure Protext using CONFIG are given in the chapter 'Configuration'. It is recommended that some experience is gained with using Protext before any changes are made.

Archimedes filenames and directories

The Archimedes system of file and directory names is different from the other systems covered in this manual and the reader should be aware of this. Filenames on the Archimedes consist of up to 10 letters and the full stop is used to separate directories when giving a full pathname.

The PC convention is that filenames consist of up to 8 letters plus, optionally, a full stop and an 'extension' of up to 3 further letters. The full stop here is part of the filename, not a directory separator.

The filename extension is used by Protext to identify particular types of file, for example the configuration file is 'protext.cfg' and printer drivers use the extension 'ppd'.

The Archimedes filenames do not have the same concept of an extension, but instead subdirectories are used to achieve the same effect. Thus printer drivers are all contained in a directory called 'ppd' and dictionaries in a directory called 'lex'. One exception is backup files which are handled differently (see 'Command mode').

Thus a reference to the dictionary 'arnoreng.lex' should, on the Archimedes, be taken as 'lex.arnoreng'.

Protext makes the use of these subdirectories almost transparent. When a file is saved in Protext with a subdirectory name included, that directory is automatically created if it did not already exist.

6. Overview of Protext

This manual introduces all the features gradually, so there is no need to read the entire manual immediately. The first part of the manual restricts itself to the word processing aspects of Protext, leaving later sections to describe the spelling checker, mail merging and utilities.

Protext is supplied on three or more discs, which contain all the program files, as well as various help files and the dictionary. These, together with this manual and the tutorial guide, provide all that is necessary to use Protext.

IMPORTANT NOTE: With the exception of the first time that Protext is used, to create a working copy of the discs, the original discs should NEVER be used. They must be retained as a backup and kept in a safe place.

If the originals are used and are damaged, such as by accidentally formatting them, or even spilling a drink on them, you will not have any back up with which to create new working copies.

We are only too well aware that manuals are items that come with programs, are intended to be used to prop the keyboard at a better angle and are only read when all else fails! We would however urge users to read the relevant 'Getting Started' chapter, which gives full details of how to create a working copy of Protext painlessly and would hope that even experienced word processor users will make the effort to read further.

About the manual

The manual is separated into a number of chapters, each of which covers one aspect of the Protext word processing system. There is no need initially to read the whole manual. In fact there would be far too much information to absorb at one go. However the relevant 'Getting started' chapter should be read thoroughly and carefully before using Protext.

The manual is written in such a way that experienced users of word processors and in particular those familiar with other versions of Protext can make use of the detailed appendices which are provided at the back of the manual and which give full technical information on all Protext commands. These will provide most of the information necessary to get started and only when some feature is not understood, will it be necessary to refer to the relevant section of the manual.

For newcomers to word processing, it is recommended that the reverse process is followed and that the appropriate section of the manual is read and studied and full use is made of the tutorial guide.

Throughout the manual it is assumed that the user has little or no experience of word processing and the main chapters have, as far as possible, been written in non technical terms, but inevitably it has been necessary to use some 'jargon' and a 'Glossary of terms' is provided at the back of the manual.

Version numbers, updates and README

Every Arnor program has a version number. The main Protext version number is displayed on the command mode banner line. The individual utility programs display version numbers when they are used. **Any queries regarding the software should be accompanied by the relevant version number, and your Protext registration number must always be quoted.**

Arnor have a policy of continual enhancement and improvement of software, and from time to time new versions of Protext will be made available. Existing users qualify for a low cost upgrade to any new version, but only if the registration card has been returned to Arnor.

Inevitably the printed documentation cannot keep up with the changes to the software, and so a text file may be supplied on the Protext disc, called 'README' giving any updates to the documentation. This file should be loaded into Protext to be read and printed. Major updates to the program will be supplied with additional pages for the manual.

How to proceed

At this point there are a number of ways in which to proceed, depending on the amount of experience already gained in the use of word processors.

Beginners to word processing are advised to read through the following chapters and to make use of the tutorial guide, in order to make the quickest progress.

Experienced users may feel that they can manage without reading through the whole manual and for their benefit the appendices give full details of all the command mode and editing commands. In spite of this, Protext has so many special features that many of them may never be discovered if the manual is not read through. The comprehensive index permits easy reference to the section describing any particular feature.

Overview of Protext

Protext gives the choice of menu or command operation. In general all functions can be carried out either by the pressing of certain key combinations or typing a command, or by selecting from a menu. Each method has its advantages - keyboard commands are often faster and more convenient to use, the use of menus avoids the need to remember the commands. Protext provides the flexibility to use menus or commands at any time - it is possible to use either exclusively or freely mix the two. The menus display the keyboard equivalents so that the most commonly used commands can be learnt, which will save editing time.

Protext has two distinct modes of operation, 'Edit mode' and 'Command mode'. Edit mode is the mode used to create the text, whilst command mode is where commands may be typed to perform operations such as saving, loading and printing are carried out. The ESC key is used to switch between edit and command mode whenever required.

When using the menus all menu options can be used from either mode. However it is not necessary to use command mode as almost all commands are also available from the menus. After selecting a menu function Protext will return to edit mode.

Menu operation

The menus are selected from a menu bar at the top of the screen. This is not always displayed but appears when the right mouse button or f3 is pressed, temporarily replacing the status line.

Some menu items are shown faintly. These are not available in the current context (for example block operations when no block markers are defined).

Some items have a tick character displayed at the left hand side to indicate when the item is selected. For example the 'Options' menu shows the current state of settings such as overwrite/insert mode.

The » character at the right of a menu item indicates that selecting this will cause a submenu to appear.

Menu operation with the mouse

Click the right mouse button to call up the menu bar. Menus are pulled down by moving the mouse pointer over the titles on the menu bar.

To select a menu item move the pointer over it and click the left mouse button.

To resume editing without selecting a menu operation simply click the right mouse button or click the left button when the pointer is outside the menu area.

***Menu operation with the keyboard** (not Amiga)*

Press the menu key, f3, to call up the menu bar. Menus are pulled down by pressing the left or right cursor keys to move on to the required menu and then pressing RETURN. Pressing the cursor keys with CTRL moves directly to the first or last menu.

To select a menu item use the down and up cursor keys until the required menu item is highlighted and then press return.

To resume editing without selecting a menu operation press ESC.

Dialogue boxes

Some menu selections (for example, find string, print) cause a dialogue box to be displayed. Various boxes and buttons will be shown which may be selected in any order by clicking the left mouse button or using the highlighted hot key. ESC can always be pressed to leave a dialogue box and cancel the operation. In some cases (such as when using a Text box) it will be necessary to press ESC twice. Dialogue boxes may contain the following items:

Text Box: a string of text is to be entered. Terminate by pressing RETURN or clicking on another box or button.
Option Button: clicking selects or deselects an option, e.g. the Find options.
Linked options: selecting some options causes others to be automatically deselected (e.g. continuous/single sheet printing).
Command Button: two types
(i) Cancel: clicking here is the same as pressing ESC, it abandons the operation.
(ii) Confirm: (GO) clicking here closes the dialogue box and proceeds with the operation using the currently selected options.

The File Selector

Whenever it is required to select a filename the file selector is used. A file may be selected by moving the cursor and pressing RETURN or by double clicking on the chosen file. The display may be scrolled sideways by clicking on the border.

When using command mode the file selector will appear if, when a filename is requested, RETURN is pressed without typing a filename.

ST *Protext may be configured to use the Gem file selector instead.*

Amiga *Protext may be configured to use the ARP file selector instead (this requires arp.library to be present).*

The menu option 'Catalogue files' will always use the Protext file selector. 'Load file' is the same as 'Catalogue files' unless the Gem/ARP selector has been configured.

Use of Amiga menus and shortcuts

Amiga *The menus support extended selection by clicking the left mouse button whilst holding down the right mouse button and by 'drag-selection'.*

The recommended right-Amiga key shortcuts are supported and are shown in the menus in the standard way. These are as follows:

Amiga-X	*Cut*	*(delete block)*
Amiga-C	*Copy*	*(copy block)*
Amiga-V	*Paste*	*(move block)*
Amiga-Z	*Undo*	*(undelete block)*
Amiga-O	*Open*	*(load file)*
Amiga-S	*Save*	
Amiga-Q	*Quit*	
Amiga-I	*Change font type to italic*	
Amiga-B	*Change font type to bold*	
Amiga-U	*Change font type to underline*	
Amiga-P	*Reset font characteristics to plain defaults*	

Note: This manual describes most features in terms of commands rather than menus as in most cases there is no difference in operation after selecting from the menu or typing the command. Any differences are noted and explained. The tutorial guide explains use of the menus.

Edit mode

Protext's editing commands are called by using certain keys on the keyboard, normally in conjunction with the CTRL key, but also the SHIFT and ALT keys. They have been carefully chosen so that wherever possible, there is an association between the task to be performed and the key used to carry it out. For example: CTRL-F to Format a piece of text; CTRL-J to turn Justification on and off.

Certain commands, such as the ones to move forward or backwards a page, or a paragraph, or to the start or end of a document have been allocated the various types of opening and closing brackets, so that they are easily remembered. Opening brackets mean a move towards the start of the document, whilst closing brackets cause movement towards the end.

Full advantage is made of the cursor keys and when used in conjunction with SHIFT, or CTRL, the effect becomes increasingly greater. For example: Using the right cursor key on its own will move the cursor one character at a time. Using it with the SHIFT key will move a word at a time, whilst with CTRL, it will move to the end of the line. (*PC: The Home and End keys may also be used to move to the start and end of a line*).

Similarly, the commands to delete make use of the two deleting keys (DEL and ←DEL), which on their own will delete one character, but when used with SHIFT will delete a word and with CTRL will delete to the beginning or end of the line.

Throughout the manual, the standard 'Protext' command keys are used to describe the commands. Protext is available for use on a range of computers and all versions use the same commands as far as possible.

It is recommended that the standard Protext commands are learnt, as versions of Protext on other makes of computer will then be usable with little or no re-learning necessary.

There is a complete summary of all the commands in an appendix at the back of the manual.

Certain features, which determine the way the text will be printed are initiated by the use of 'Stored commands'. The number of lines to a page, where a new page break should be made, the width of the text and many other features can all be controlled with stored commands.

A stored command is an instruction which is inserted into the text and which is acted on when printing takes place, rather than being printed. Stored commands take the form of a 'greater than' symbol (>) in column 1 of the text, followed by two characters which define the command. For example: '>PA' will cause a new **PA**ge to be taken and '>PL' will define the **P**age **L**ength.

Additionally, features of the printer, such as the style of printing, whether bold or underline, are controlled by the use of printer control codes. These are single characters which are inserted into the text and which Protext will recognise when printing and send to the printer as the codes that the printer requires to initiate the function required. Printer control codes are inserted by pressing CTRL-X and then a letter key. Printer control codes appear on the screen as the letter in inverse video. Again, the default settings chosen have been selected for ease of use. For example: '**i**' is used to initiate **i**talic printing and '**b**' for **b**old printing.

A brief description of the typical procedures which might be followed to create and print a letter is also provided in an appendix at the back of the manual.

Command mode

Command mode can be recognised by a broad band about two thirds of the way down the screen and immediately beneath this is the command mode prompt 'A»' *(Amiga/Archimedes: just '»')*, followed by the cursor. (Hard disc users will normally find the prompt will contain the letter 'C'. The letter indicates the currently selected drive). Whenever the prompt and the cursor are visible, Protext is waiting for a command. Commands are words which are typed in as instructions and may be followed by one or more parameters, depending on the command.

There are a considerable number of commands, which are covered in detail in the 'Command mode' chapter. Commands and parameters may all be entered on the same line, with the parameters being separated from the command by a space. If more than one parameter is specified, they may be separated either by a space or a comma (,).

Any commands which need further parameters can be used just by typing the command and Protext will prompt for the parameters. There are, however a number of commands which, optionally, may have parameters specified, in which case the parameter should follow the command, separated by a space.

Once the command has been specified, the RETURN key should be pressed and the command will be carried out and when completed, the command mode prompt will return.

Special keys

There are various different types and layouts of keyboard in use, but they all have the various special keys that Protext uses though the keys may be located in completely different positions. As far as possible the keys used for commands have been chosen so that they are positioned in convenient places on all keyboards.

PC	*The CONFIG utility program contains an option to use alternative keyboard drivers. See 'Configuration' for details.*

One particular source of confusion is caused by keyboards that have two keys marked '←' - one of which moves the cursor left and the other moves the cursor left and deletes the previous character. This table summarises the differences - the left column is the name that will be used for the key throughout the manual.

ALT	-	means the key marked 'Alt' or 'Alternate'.
CTRL	-	means the key marked 'Ctrl' or 'Control'.
DEL	-	means the key marked 'Del' or 'Delete'.
←DEL	-	means the backspace key which is usually marked '←' but marked 'BACKSPACE' on the Atari ST keyboard.
←	-	means the 'cursor left' key to the right of the keyboard. Not to be confused with '←DEL'.
INS	-	means the key marked 'Ins' or 'Insert'.
TAB	-	means the key on the left of the keyboard, often marked with two opposite pointing arrows.
keypad+	-	the plus key on the numeric keypad, as opposed to the plus key on the main keyboard.
keypad-	-	the minus key on the numeric keypad.

Conventions used

Protext uses a 'standard set' of commands which, as far as possible, are common to all versions of Protext and these are used to describe the functions.

Edit mode commands

Throughout the manual a consistent form is used to describe the various commands, as follows:

CTRL-I - means the key marked 'CTRL' and the key marked 'I'. Wherever a hyphen is used between them, it means that the first key should be held down whilst the second key is pressed. Most of the editing commands take this form.

CTRL-V T - means that the 'CTRL' and 'V' keys should be used as described above, then released and the 'T' key pressed. Note that there is no hyphen between the 'V' and the 'T'.

CTRL-SHIFT-S - means that all three keys should be pressed at the same time. This sort of command that requires more than two keys to be pressed at a time is rarely used and at least two of the keys are always adjacent to each other.

CTRL-(- means that the 'CTRL' key and the key which has the '(' on it are pressed together. It does NOT mean that SHIFT is required as well. The '(' is merely being used for ease of remembering its function.

Sticky shift keys

Protext may be configured so that it is not necessary to press CTRL, SHIFT or ALT at the same time as another key. To do this use CONFIG and set the 'CTRL/ALT/SHIFT lock' option. Then, any command such as CTRL-I may be entered by pressing and releasing CTRL, then pressing I. Capital letters may be entered in the same way using SHIFT, which is one less key press than turning CAPS LOCK on and off.

Command mode commands

Command mode commands are always shown in upper case, though when they are being entered into the computer, they may equally well be entered in upper or lower case. Similarly, when entering filenames to LOAD or SAVE a file, even though they may be shown in upper case, lower case is acceptable and Protext will automatically convert them to upper case if required.

Stored commands

Stored commands are always shown with their two letter name in upper case, but again, upper or lower case is acceptable.

Sections specific to one system only

As Protext is extremely compatible between the different computer systems this manual covers all versions. Inevitably, though, there are a number of differences due to the different operating environments. To avoid confusion, sections covering these differences are indented and printed in italic text and a note in the margin indicates to which system the section refers.

Editing large files

Protext is capable of handling large files very efficiently and the only limit on the size of the files which can be edited is the capacity of the disc drives. It must be remembered that very large files cannot be totally loaded into memory at one time, and as editing continues and progress is made through a long document, Protext will automatically save parts of the document as temporary files.

As a result, when floppy discs are being used, it is preferable to start editing a large file with as empty a disc as possible.

Note: It should also be remembered that there must be sufficient space on the text disc to save the amended file when editing is completed. By default, Protext renames the original version of the file so that it has a '.BAK' suffix. CONFIG contains an option to turn off this facility, in which case the old version will be erased. When working with large files, it may be beneficial to turn the BAK option off, but it should be appreciated that there are inherent dangers in doing so, in that there is no longer a back up of the file to use if necessary.

In the event that the document becomes so large that there is no room left for the temporary files to fit, a 'Disc full' message will be issued. If this happens, it will usually be possible to delete one or two files from the disc or drive to make room, before continuing. For example the disc might have copies of the help files on it, in which case deleting these would give more space. Once this situation has been reached, it is worth considering whether the document should be broken down into smaller parts if further editing is required.

Warning: Whilst using Protext do not delete files called PROTEXT.L, PROTEXT.T and similar. These are temporary files that Protext is using.

Other than the points mentioned above, editing of large documents is exactly the same as editing any other document. It should also be remembered that the CTRL-[and CTRL-] commands move to the start and end of the text in memory, not the start and end of the whole document. With a small document this will be the same thing, but if the start or end of a long document is required, then CTRL-[or CTRL-] should be pressed a second time.

Are large documents necessary?

Even though Protext can handle 'unlimited size' files, this is perhaps a suitable place to consider whether it might be more convenient and efficient to work with a number of smaller files. Rarely is there any need for a long document to be in one piece. For example: a book is broken down into a number of chapters, which would normally make suitable places to split the manuscript.

Whilst it may appear that there are advantages to being able to work on one long document, so that it can all be viewed and edited at the same time, there are a number of points which should be considered.

1. In the event of a catastrophe, such as a power failure, or accidentally deleting a file from a disc, if the text is in one long document, the complete document may be lost.

2. It is not always possible to have the whole of a large document in memory at the same time and as progress is made forwards and backwards through the document, parts of it have to be saved to temporary files and other parts loaded. Since this involves the disc drive this is slower than if the whole file is loaded into memory.

3. It is usually easier to locate specific sections of text in a smaller file.

4. Usually only a relatively small part of a document will actually be worked on at a time and it is considerably quicker to load, save and format smaller files.

5. Protext has many features, such as 'multiple file editing', 'printing to the screen' and the ability to link files together whilst printing, which make it easy to view, move sections of text around and print a number of files as one file.

Note: these considerations diminish in importance as more memory and/or disc space becomes available. *However on a PC the amount of text that can be held in memory from one file is limited to 64K and so there is a speed cost when editing files larger than 64K.*

Graphics and text modes *PC only*

PC Protext will work in most text or graphics display modes. To change between text and graphics press ALT-G. This will work in conjunction with ALT-F to change the number of lines displayed, allow the selection of 25 line text, 43/50 line text, 25 line graphics and 43/50 line graphics. These display options can also be selected using CONFIG so that a given setting is always used.

If the display adaptor being used supports other modes, for example VGA modes with more rows or columns, then Protext will probably be able to use these modes. Simply select the required mode using the VGA system software and Protext will in most cases automatically recognise and use the available lines and columns.

Graphics mode is required for the following:

Display of italics, bold and underlining (except monochrome text mode which shows only bold and underlining). Display of accents on characters, other than those characters included in the standard character set (see appendix). Display of certain other symbols such as o-slash and l-slash.

Text mode has two advantages - screen operations are faster and it uses slightly less memory. In practice the speed is only significant when the screen is scrolling.

7. Entering and Correcting Text

Once Protext has been loaded, three lines containing information about the state of the program will be visible at the top of the screen. These are the 'Status lines', the contents of which will be explained later. There is also a thin horizontal line which marks the end of the text.

At this stage the program is in edit mode, which is used for all entry and correction of text. Pressing the ESC key will put the program into Command Mode, which is described in detail later. A line will appear about two thirds of the way down the screen containing further information. Pressing the ESC key at any time will return to edit mode, as will using the menus.

On screen help

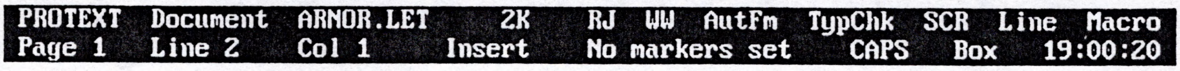
There are two distinct types of 'on screen help' available. One is for help whilst in edit mode and this is contained within the menus - each menu option with an equivalent keyboard command shows this to the right of the menu. Various abbreviations are used here - '^' means press CTRL, 's' means press SHIFT and 'a' means press ALT. For example:

^I	means hold down CTRL and press I
s^S	means hold down SHIFT and CTRL and press S

The second form of help is provided by the help menu and provides explanatory text on various topics and commands.

The status line

The status line at the top of the screen shows the following information:



```
PROTEXT  Document  ARNOR.LET     2K    RJ  WW  AutFm  TypChk  SCR  Line  Macro
Page 1   Line 2    Col 1      Insert   No markers set    CAPS    Box   19:00:20
```

First line from left to right -

'PROTEXT'; 'Document' or 'Program'; The name of the current document; the size of the document; 'RJ', if right justify is on; 'WW', if word wrap is on 'AutFm', if auto-reformatting is on; 'TypChk', if spell check whilst typing is on 'SCR', if scroll lock is on; 'Line', if line drawing is enabled; 'Macro', if currently recording a macro.

Second line from left to right -

The current page, line and column number (or the character, line and column if page mode is turned off); 'Insert' or 'Overwrite'; Any block or place markers that are set; Whether CAPS and/or NUM lock are set; 'Box', if box mode is on; the current time.

In program mode the text formatting options are not shown and the top line additionally shows whether 'auto-indent mode' is enabled.

Entering text

Once in edit mode a flashing cursor is positioned beneath the status lines and anything that is typed at the keyboard will appear on the screen at this position and the cursor will be moved forward one position.

As practice, a section of text might be entered by copying a paragraph out of this manual. There is no need to press RETURN at the end of each line, the words simply being typed in, one after the other. At the end of a line the cursor will automatically move onto the next line. When the end of a paragraph is reached, the RETURN key should be pressed.

Any mistakes made whilst typing, which are noticed at the time, may be corrected by pressing the ←DEL key, which will cancel the last character entered.

The cursor can be moved around the screen by pressing the four cursor keys (those with arrows on them). By using these keys, text may be entered at any position. The cursor moves one line or column for each press of a cursor key. Holding a cursor key down will make the cursor move continuously. Releasing the key will stop the cursor moving.

The cursor cannot be moved past the end of text (the thin horizontal line on the screen). To position the cursor further down, the end of text must be moved down by positioning the cursor at the end of the text and pressing RETURN as many times as required.

Upper and lower case

Initially the letter keys produce lower case letters, unless SHIFT is pressed at the same time. If Caps Lock, is pressed, upper case letters are always produced, and this is indicated on the status line.

Protext has commands to change the case of a letter or word. To make a letter upper case, press CTRL-/ when the cursor is on the letter. This command only affects letters, so the cursor can be moved quickly over a line to convert all letters to upper case by holding down CTRL-/. Similarly, CTRL-\ will convert upper case letters into lower case. Pressing SHIFT at the same time as these commands will convert all characters from the cursor to the end of the word.

Deleting

The ability to move the cursor around, permits the correction or alteration of text anywhere on the screen. The cursor should be positioned on the letter to be changed and the DEL key pressed. This will remove the letter at the cursor position, and move the rest of the line to the left. As many letters as required can be deleted in this way. If the new letter is now entered it will appear on the screen and the rest of the line will move back to the right. Alternatively, pressing ←DEL will remove the character to the left of the cursor and the text will again move to the left to fill the gap. Repeated pressing of either DEL key will cause further characters to be deleted.

If extra text is to be inserted, the cursor should be positioned where the first new character is to be added and the new text entered.

Just as a character can be deleted, so can a word. Pressing SHIFT and DEL when the cursor is at the start of a word will make the word disappear. If this is done when the cursor is in the middle of a word, only that part of the word at and to the right of the cursor position will be deleted.

Similarly, pressing SHIFT and ←DEL will remove the word to the left of the cursor, or if positioned in the middle of a word, the characters to the start of the word.

CTRL-←DEL will delete all text from the character on the left of the cursor to the start of the line and CTRL-E will delete all text from the cursor to the end of the line.

Finally there are two further useful deleting commands. SHIFT-CTRL-E deletes all text up to the end of the sentence - that is up to and including the next full stop, question mark or exclamation mark as well as any following space. SHIFT-CTRL-←DEL will delete back to the start of the sentence.

CTRL-f3 will delete the whole line. The line is removed from the document and the remainder of the text moved up a line.

Note: Pressing CTRL-←DEL followed by CTRL-E will delete all the text from a line, but will not remove the empty line from the text, unlike CTRL-f3, which will remove the blank line as well.

Inserting

To insert a new blank line into the text, CTRL-I should be used. The cursor will remain where it is and all text from the current line to the end of the document will be moved down a line.

Swapping characters

A common typing mistake, especially when typing quickly, is to type two letters the wrong way round, e.g. 'wrod' instead of 'word'. The CTRL-A (Alternate characters) command will put this right. The cursor should be positioned on the first of the two offending characters (on the 'r', in the above example) and CTRL-A pressed. The two characters will then be exchanged.

Undeleting all or part of a line

Protext maintains a temporary store (called a buffer) which always contains the most recently deleted section of text. If a line or part of a line, more than three characters long, is deleted, the deleted text will be saved in the buffer. If a section of text has been accidentally deleted, it may be restored by pressing CTRL-U.

This command can also be put to good use for moving lines or parts of a line to a different position in the text, though this is not the purpose for which it is really intended. The text to be moved should be deleted using one of the word or line delete commands and the cursor moved to the position in the text where the deleted text is to be placed. Pressing CTRL-U will then restore the text at the new location.

Note: Only the text removed by the last delete command will be stored in the buffer. The buffer contents are changed when a section of text containing 4 or more characters is deleted.

Insert and Overwrite mode

Initially Protext, by default, is in insert mode and the word 'Insert' is displayed on the status line at the top of the screen to indicate this. This means that when text is typed, the rest of the text on the line is moved along to make room. This is the mode that is preferred by most people for text entry.

Pressing CTRL-TAB (or the Insert key) will change the status line to 'Overwrite'. Selecting overwrite mode can make certain editing tasks easier and the effect of using it means that if the cursor is positioned over an existing piece of text and new text typed in, the existing text will be replaced by the new text, unlike insert mode, where the existing text would be moved to the right.

If an extra character needs to be inserted whilst in overwrite mode (for example if replacing a word by a longer word), this can be done by pressing CTRL and the space bar which will move the text to the right to make room.

Moving the cursor more rapidly

So far the cursor has been moved by a character at a time, but there are also various ways to move the cursor more quickly. These are as follows:

(a) Pressing SHIFT-→ or SHIFT-← will make the cursor jump a word to the start of the next (or last) word. This feature is useful for moving more quickly to a word which needs correction.

(b) Pressing CTRL-← or CTRL-→. This moves the cursor to the beginning or end of the line. (*PC: The Home and End keys may also be used*).

(c) Pressing SHIFT-RETURN or CTRL-RETURN. This moves the cursor to the beginning of the next line, without causing a new line to be inserted, which would happen if the RETURN key was used on its own.

(d) Pressing CTRL-↑ or CTRL-↓. This moves the cursor up or down almost a full screen, but with a few lines overlap so that the context may more easily be followed. By holding down CTRL-↑ or CTRL-↓ the text can be rapidly scanned. (*PC: Similar functions are performed by PgUp and PgDn*).

(e) Pressing CTRL-[or CTRL-] moves the cursor to the beginning or end of the text in memory. Pressing the same keys a second time will move to the beginning or end of the document. (*PC: CTRL-Home and CTRL-End will take the cursor straight to the beginning or end of the document*).

(f) Pressing CTRL-< or CTRL-> moves the cursor backwards and forwards by a paragraph at a time.

(g) Pressing CTRL-(or CTRL-) moves the cursor by a page at a time. This is a page as it will be printed, not a 'screen page'.

(h) Pressing CTRL-@ [or CTRL-@] will move to the opening or closing block markers, if set. See chapter on Cut and Paste editing for details of block markers.

(i) Pressing CTRL-@ L or CTRL-@ R will move to the left or right margin on the current line.

Note: There is a difference between using CTRL-@ L or CTRL-@ R and CTRL-← (which will always move to column 1) and CTRL-→ (which will always go to the end of the text on that line, which may be somewhat less than the right margin).

(j) Pressing CTRL-f6 or CTRL-f5 will go to the next or previous marker in the document. See 'Place markers' and 'Multiple markers'.

(k) Pressing CTRL-L moves the cursor back to the last position. This is particularly useful if the cursor has accidentally been moved to another part of the text by using an incorrect command. CTRL-L will return the cursor to the position where it was before the incorrect move was made. It will only have any effect if the cursor has been moved with one of the 'jump' commands. Moving the cursor a single space, or line, at a time will not affect the use of CTRL-L and it can still be used to return to the original position from which the last jump was made. With care, this facility can be put to good use, by permitting a jump to another part of the text, where one or two alterations or additions may be made, before pressing CTRL-L to return to the original place in the text.

Moving the cursor with the mouse

The cursor can be moved to any position on the screen by moving the mouse so that the mouse pointer or cursor is at the required position and clicking the left mouse button.

Clicking on the second status line at the top of the screen will scroll the text up (the same as pressing CTRL-↑).

Clicking on the bottom line of the screen will scroll the text down (the same as pressing CTRL-↓).

Moving to a specified page, line or column number

Pressing CTRL-G will result in a dialogue box appearing offering 'Line', 'Column' and 'Page'. Entering 'P' (or clicking on the box next to the word 'Page'), followed by a number will move the cursor to the start of that page. Similarly, 'L' plus the line number will move to the line. Prefixing a number with 'C' will result in the cursor moving to the appropriate column. Moving to a column is particularly useful when wishing to move a long way across the screen, say to create a wide ruler for some special reason, and is considerably quicker than holding the right cursor down.

If no letter prefix is used, Protext will treat this as meaning a line number.

Note: The line number used by this command may be different from the line number on the status line. This is because when in page mode, the line numbers shown on the status line take stored commands into account, whereas this command uses the absolute line number in the text. Typing CTRL-P will switch 'page mode' off and the display will then show the line number used by CTRL-G.

Place markers

A place marker can be put anywhere in the text and is similar in use to a book marker. Ten place markers can be set, numbered 0 to 9. A place marker is set by pressing CTRL-@ followed by the number. When a marker has been set, it will appear in the text as the number in inverse and will be shown on the status line, so that by looking at the status line it is easy to see which markers are available. Once a place marker has been set, it can easily be returned to at any time by repeating the CTRL-@ command with the same number.

In addition to using CTRL-@ and the number to find a place marker, it is possible to jump from one to the next in the document by using CTRL-f6 to move on through the document, or CTRL-f5 to move backwards. Using these commands will find the next or last marker in the text. All types of markers (place, multiple and block) will be found. They are not treated numerically, but are found in the order in which they occur in the document. As an example of the use of a place marker, suppose a long file is being edited and something needs to be added at the top of the text. A place marker can be set and CTRL-[pressed twice, to move to the top of the text, and after making the addition, CTRL-@ and the place marker number used to move back to the place marker.

Note: Place markers are saved with the text and will be restored when the file is reloaded. If one file containing markers is merged into another, duplication of markers may occur. The duplicated markers can be deleted in the same way as any character.

Multiple markers

In addition to the 0 to 9 place markers, Protext also has another special marker. This is known as a 'multiple marker' and is inserted into the text in a similar way to the place markers, the difference being that whereas only one of each place marker can be set in a document, as many of the multiple markers as required may be set. CTRL-@, followed by pressing the key with a question mark (?) on it will set a multiple marker and this will appear in the text as an inverse question mark. Multiple markers are saved with the text.

An example of how a multiple marker may be used would be where a standard document was created, which required different items of text to be inserted in a number of set places. If a document was created with a multiple marker everywhere that required insertion of text at a later date, when the file was loaded the next time, it would only be necessary to use CTRL-f6 to go to the first occurrence of a marker and enter the relevant text. Repeated use of CTRL-f6 would find subsequent occurrences. All that would then be required before printing would be to use the FORMAT command to reformat the document. If the multiple markers are positioned in the text by use of tabs, neatly tabulated lists of items are easily created.

Special markers

Three other types of marker are provided for special uses. The first is the 'spell ignore marker', which is used to prevent a word being spell checked. This is entered by typing ALT-=. See 'Spelling Checker' chapter for details.

The other two are 'index markers' and are used to mark words for indexing. ALT-W is used to mark a single word, ALT-P is used at the start and end of a group of words to mark a phrase. See 'Index and Contents' for more details. and later.

Scrolling

When the text fills the entire depth of the screen, typing further text will cause the screen to scroll up. That is, the top line will disappear and the rest of the screen will move up one line to make room for a new line at the bottom of the screen.

In the same way the text will scroll if the cursor reaches the bottom of the screen but there is more text to come, or reaches the top of the screen when the text has previously scrolled. This is known as vertical scrolling, and is essential for editing text that is longer than a few lines.

Protext has commands to force the screen to scroll either up or down at any time. This is done by pressing SHIFT-↑ or SHIFT-↓. The cursor will stay on the same line, but the whole text will scroll by one line. This feature is useful if a line is to be edited and it is desirable to see the text beneath or above.

There is another form of scrolling, called horizontal scrolling, which happens automatically when the cursor is moved beyond the right hand limit of the screen. If this is done the text will scroll to the left. This means that the text on the left of the screen will start to disappear as the cursor is moved further to the right of the screen. Horizontal scrolling allows text to be entered in lines that are longer than the screen width. This can be confusing at first and so is best avoided initially. If horizontal scrolling occurs, any of the commands which move the cursor to the left may be used to scroll the text back, or SHIFT and RETURN may be pressed together, which will position the cursor at the start of the next line.

Splitting and joining lines

Lines will often require splitting, or joining together. This is very easy in Protext. There are two different methods of doing this, depending on whether 'Insert' or 'Overwrite' mode is in operation.

To split a line whilst in Insert mode, the cursor should be moved to the character which is to be the first on the new line and RETURN pressed. To join two lines, either move to the end of the first line and press DEL, or move to the start of the second line and press ←DEL. The text on the second line will then move up and join onto the end of the text on the first line.

If in overwrite mode, CTRL-* will split the line at the cursor and CTRL-+ will join the next line to the end of the current line.

Note: The asterisk, '*', in the above command is the one found on the number 8 key towards the top of the main keyboard. On most keyboards the asterisk is duplicated on the numeric keypad and this should not be used as the key for CTRL-*.

8. The File Selector and Disc Utilities

This chapter describes the Protext file selector and a number of disc functions which are provided from the file selector.

When you select 'Catalogue files' or 'Load file' from the File menu, you are initially presented with a list of all files in the current directory, together with their size. The files are all listed vertically and several columns are used if there are too many files to fit in a single column.

A number of commands are listed in the panel at the bottom of the screen. Commands are selected by pressing the key or key combination shown inside square brackets or by moving the mouse pointer inside the square brackets and double clicking.

The file selector may be abandoned at any time by pressing the ESC key.

Movement around the directory listing

The ↑ and ↓ cursor keys may be used to move the highlight bar up and down the list of names.

The ← and → cursor keys may be used to move from column to column if more than one column of files is displayed.

CTRL-↑ and CTRL-↓ move to the top and bottom of the column respectively.

CTRL-← and CTRL-→ move to the left and rightmost columns respectively.

CTRL-[and CTRL-] move to the first and last entries in the listing respectively.

Alternatively the mouse may be used to select a file. Simply move the pointer on to the required file and click. If all files are not displayed, click the mouse at the left or right edge of the screen and more entries will appear.

Selecting a file or directory

Pressing RETURN selects the currently highlighted file or directory.

If the highlighted item is a directory, that directory will be selected as the current directory and all files matching the 'file specification' will be displayed.

Note: Except for the root directory, the first entry in the list is always <PARENT> <DIR>. Selecting this entry and pressing RETURN selects the parent directory.

If RETURN is pressed whilst a filename is highlighted, Protext will load that file (or, for example, merge a file if the file selector has been called by a merge file command).

Viewing in a different format

Filenames may be displayed in three different forms and these may be toggled, in rotation, using the TAB key.

Brief - Filename, with file size in Kilobytes, displayed in up to four columns (*Amiga: two columns*). This is the initial state.

Expanded - Filename, with exact size in bytes, date and time of creation.

Attribute - Filename, with exact size in bytes, date and file attributes. 'attributes' indicate the status of the file. They are displayed as the appropriate letter, if set, otherwise as a dash. The file attributes letters and the meaning when set are:-

R	read-only
H	hidden
S	system
V	volume label
D	directory
A	archive bit set

Amiga Note: only 'R' and 'D' apply. 'S' is the script bit.

Viewing different files

By default, all files are displayed, but this may be changed to restrict the display to a set of files. Two commands are provided for this purpose.

[A] Show all selects all files for display.

[F]ile spec. allows selection of a different file specification.

Any file specification may be entered and only files matching the specification will be displayed.

Note: Pathnames may NOT be included. If a different directory is required, this should be selected by using one of the directory commands. If a different drive is required, the [D]rive change option should be used.

Changing drives

In addition to being able to select a directory by highlighting it and pressing RETURN, a command is provided to select a different drive.

PC and ST

[D]rive change *requests the drive letter to be changed to and displays all files matching the file specification in the current directory on the chosen drive.*

Amiga and Archimedes

[D]ir change *requests the drive name and directory to be changed to and displays all files matching the file specification in the chosen drive/directory. 0 or 1 may be pressed to select the first two floppy drives.*

Creating a new directory

[N]ew dir. requests a directory name and creates a new directory. A full pathname may be given to create a directory anywhere on the disc, or just a name to create a subdirectory within the current directory.

Viewing files

A command is provided which lets you look at the contents of a file. This command is intended for use with text files and provides a convenient method of viewing the contents of files without loading them.

[T]ype	displays files on screen.

If Protext's 'scroll lock mode' (*PC/Archimedes - Scroll Lock key. Amiga/ST - CTRL-HELP*) is turned on, the listing will be displayed in 'pages' with a prompt to 'Press a key to continue' at the end of each 'page'.

Erasing, Renaming and Copying files

Protext also provides commands to erase, rename and copy files.

[E]rase	erases the highlighted file after asking for confirmation. If a directory is highlighted then it is removed only if it is empty.
[R]ename	requests the new name required and renames the highlighted file to the new name. Note: Only a filename (with or without a filename extension) may be specified. It is not possible to rename a file to a different directory.
[C]opy	requests the 'From' filename, which may include a pathname, if required, then requests the filename, or pathname to which the file is to be copied. The destination ('To') may be a pathname, in which case the file will be copied to the directory specified in the pathname, saving the file with the same name as the original. If a filename is also specified as part of the 'To' pathname, the file will be copied to the specified directory and saved with the filename specified.

File Protection

Pressing CTRL-R will toggle the Read/write file attribute of the selected file between Read/write and Read only. When a file is set to Read only, it is shown in the listing marked with an asterisk.

Files which are set to read only cannot be overwritten by other files of the same name, or with more recent copies of the same file.

Directory Tree

Protext also provides the facility to view the structure of the disc, in the form of a directory tree, or map. This command is of particular use to those people using hard discs.

[M]ap	provides a visual representation (map) of the directory tree structure for the current drive.

When you select [M]ap, the display changes to show the complete directory tree, or map, of the currently selected drive/disc. Also the amount of free space on the disc is displayed near the bottom of the screen.

Note: There is a slight delay the first time that this command is used, whilst Protext reads the complete directory structure into memory. This occurs the first time and after changing drive.

All the directories on the disc are displayed down the left hand side of the screen, with the root directory at the top. A line extends down the left of the display, with branches off it to each directory. Any sub-directories of these directories are similarly linked to their parent directory by a line and are offset to the right. This process continues to as many levels of sub-directories as necessary.

To the right of the directory names are two columns, one of which displays the total size (in Kilobytes) of all files within that directory. Note that the figure includes ALL files within that directory, including those in any sub-directories. This is particularly useful, as you can tell at a glance the total amount of space taken up by a sub-directory and its 'children' just by looking at the size alongside it.

The second column provides a count of the number of files in the directory. Again, like the file size entry, the number of files includes all files in sub-directories.

The ↑ and ↓ cursor keys may be used to move the highlight bar up and down the tree and CTRL-↑ and CTRL-↓ may be used to move a screen at a time. Alternatively the mouse may be used to click on the required directory (scroll up or down by clicking on the top or bottom border).

Pressing RETURN (or double clicking) will select the highlighted directory and display all files matching the current file specification.

If you have a complex directory structure, this command is often the quickest way to select a different directory. It is also useful as a means of getting an overall view of the disc directory structure, particularly with hard discs.

9. Saving Text and other Simple Commands

Operations such as saving, loading or printing may be carried out either from the menus or from command mode. This section explains the use of command mode, which can be entered at any stage of editing simply by pressing ESC. Pressing ESC a second time will return to edit mode.

When ESC is pressed, the bottom part of the screen will clear and the command mode banner line will appear, displaying some status information such as the current directory. The cursor will be positioned next to a '»' symbol. This symbol is the 'command prompt' and indicates that commands may be entered.

The most important commands are described briefly here. Full details of these and all the other available commands are given in the 'Commands' chapter.

Saving a document

To save a document, put Protext into command mode and type:

```
SAVE                    and press RETURN
```

The message 'SAVE filename:' will be displayed. Type a name for the document and press RETURN. The file will be saved.

The second and subsequent times the document is saved Protext will have remembered the name and will display it when SAVE is entered. Just press RETURN twice to retain the same name. To save the file under a different name press ESC at this point and type the new name.

Loading a document

To load a previously saved document, put Protext into command mode and type:

```
LOAD                    and press RETURN
```

The message 'LOAD filename:' will be displayed. Type the name of a previously saved document. The file will be loaded. Press ESC to edit the file.

Printing a document

To print the document that is currently being edited, put Protext into command mode and type:

```
PRINT                   and press RETURN
```

Assuming the printer is correctly set up, the document will start printing immediately. If it does not, refer to the printer trouble shooting section later in the manual.

Clearing the text

To clear the text from memory in order to commence a new document, put Protext into command mode and type:

```
CLEAR                   and press RETURN
```

Protext will issue a warning if an attempt is made to clear a document that has not been saved. If a new document is loaded into memory, then any existing text will automatically be erased first.

Listing the files saved on disc

To list the names of files saved on the disc, put Protext into command mode and type:

```
          CAT                     and press RETURN
or press  f2
```

This will display the names and sizes of files.

Counting the words

To count the words in the text, put Protext into command mode and type:

```
COUNT                   and press RETURN
```

Finding text

To locate some text in the document, put Protext into command mode and type:

FIND and press RETURN

The message 'FIND string:' will be displayed. Type the word or words to be found and press RETURN. The message 'Enter options (A,B,C,G,S,W,n):' will appear. To search from the current cursor position just press RETURN, to search the whole document type 'G' and press RETURN.

Quitting Protext

To finish using Protext and return to the operating system in order to run another program, put Protext into command mode and type:

QUIT and press RETURN

If a file has been left unsaved a warning message will appear.

It is advisable to use QUIT before switching off the computer because Protext will then tidy up by deleting any temporary files it has created.

10. Cut and Paste Editing

This is where real word processing begins. Protext allows any section of text to be moved or copied to any other part of the text. This is often called 'cut and paste' editing.

This chapter will describe the ways to use block mode editing. A block of text is any continuous section of text. It may be of any length and may start at any position in the document and finish at any position. When in block editing mode, all text between these two points will be manipulated in whatever way is chosen.

Defining a block

The first requirement is that the block of text is marked with block markers. The cursor should be moved to the start of the section of text and CTRL-Z or f9 pressed. This will set a block marker. The marker will be indicated on the screen by an inverse video square bracket. The cursor should then be moved to the end of the section and CTRL-Z pressed again, to set a second marker. The block has now been defined and the extent of the block will be indicated by highlighting it in a different colour or in inverse. An opening square bracket is the start marker, a closing square bracket the end marker. When markers are defined, this will be indicated on the status line, where the message 'No markers set' will be replaced by 'Markers [ ]', showing that both the start and end markers are set.

The markers can be set in either order, and can be at any position in the text. The first marker set will be displayed as an opening bracket, but if the second marker is positioned earlier in the text than the first marker, this will change to a closing bracket. If the marker is put in the wrong place, pressing CTRL-Z again while the cursor is still on the marker will remove it. Either or both block markers can be cleared at any time, by pressing CTRL-K.

Often a block will consist of a number of complete lines. To define a block like this, the first marker should be positioned at the start of the first line, and the second marker at the start of the line following the last line of the block.

Defining a block with the mouse

A block may also be defined by moving the mouse pointer to the start position, pressing and holding down the left mouse button, moving the pointer to the end position and releasing the button. When this 'dragging' method is used the block markers will not be displayed, the block will just be highlighted. The block may be cleared by double clicking the left button.

Moving or copying a block

Once a block has been defined, it can be moved to any point in the text simply by moving the cursor to the required position and pressing CTRL-M. The markers will move with the text. The cursor must not be within the block at the time; if it is, an error message will be displayed in an alert box. Pressing ESC will return to edit mode and the cursor can be moved to the correct position.

The block can also be copied, leaving the original text intact. This is done by pressing f10. The markers will be moved with the block, which makes it easy to see clearly where the new copy of the block is and also to copy the block again if required. The cursor must not be within the block.

Deleting a block

The section of text to be deleted must be defined in the usual way. Pressing CTRL-DEL will delete the block. If the block is larger than the size of the 'undelete' buffer, (see below) a beep will sound and a warning dialogue box will be displayed, requesting confirmation that the block is to be deleted. The block will only be deleted if 'Y' is selected.

Undeleting a block

If a block of text is accidentally deleted, it may often be recovered by use of the CTRL-U command. When text is deleted, Protext retains the deleted block in a buffer and CTRL-U will restore it to the document.

Note: The size of the undelete buffer may be adjusted by use of the configuration program CONFIG if it is found to be too small. If a block of text which is too large for the buffer is to be deleted, a warning will be given, with the option to continue. If 'Y' is selected, the block will be deleted and the buffer will be filled with as much of the text as it can hold and the remainder will be discarded.

Note: A block can only be restored until such time as further text is deleted, after which time the buffer will contain only the most recently deleted text.

Alternative keys for block commands

f9 duplicates the function of CTRL-Z (set or clear block marker).
CTRL-f9 duplicates the function of CTRL-K (clear block markers).
CTRL-f10 duplicates the function of CTRL-M (move block).
CTRL-INS or CTRL-Insert duplicates the function of f10 (copy block).

Cut and paste editing - Box mode

The block mode cut and paste facilities already described only operate on continuous sections of the text but the Box mode editing facilities described in this chapter extend this to operate on any section of text that can be defined by drawing a rectangle on the formatted text. A block defined in this way is termed a 'box'. There is one new command used for box mode.

CTRL-B is used when in edit mode to toggle box mode on or off as required. Normally one would leave box mode turned off, except when needed.

Defining a box

A box is defined by first typing CTRL-B to enter box mode and the message 'BOX' will appear on the status line to indicate that box mode is in use. The cursor should be moved to the top left corner of the (imaginary) box and CTRL-Z or f9 pressed to set a marker, in the same way as described in the previous chapter. The cursor should then be moved to the bottom right corner of the box and CTRL-Z pressed again, to set a second marker. Whereas in block mode only a closing block marker appeared, this time markers will appear along both the left and right hand sides of the box to mark its limits.

Moving or copying a box

A box can be moved or copied, using the same commands (CTRL-M, f10) that were used for block commands, to move or copy the box. The cursor should be positioned where the top left corner of the box is required and then the appropriate copy/move command used.

Note: When using box mode, the effect depends on the insert/overwrite mode setting. If insert has been selected and a box is positioned over existing text, that text will be moved to the right to make room for the box of text. If overwrite mode has been selected, the box will erase the text underneath.

Deleting a box

Box mode delete can be used in one of two ways, depending on whether insert or overwrite mode is in operation at the time the box delete command is used. Once a box has been marked out for deletion, CTRL-DEL may be used to delete the box. If insert mode is in operation, the effect of this will be that once the box has been deleted, any text to the right of the box will move across to the left to close the gap.

Alternatively, if overwrite mode is selected, using CTRL-TAB, before the delete command is used, the box of text will be cleared, but the area where the box used to be will remain as a blank area.

Very effective layouts can be achieved by the use of box mode. One example would be to create a page with two columns of text, along the lines of a magazine page. This is simply done by creating the document, formatting it with a width of about 35 characters and then using box mode to move the second half up and to the right, positioning it alongside the first part of the text. This should be done immediately prior to printing, as any further reformatting might destroy the formatting of the two columns.

See also 'Newspaper column printing' in the 'Printing' chapter.

Another use for box mode which can save a considerable amount of time is to remove surplus columns of figures from text. A box can be marked round the columns which are not required and the box can be deleted using either overwrite or insert mode, depending on whether the remaining text should be moved across or not.

Note: When a box move is carried out and the box contains tabs, it may sometimes appear that the justification has been destroyed. The justification can be restored either by inserting an extra tab marker on the ruler line, or by replacing the tab, in the text, with spaces.

Note: When Protext automatically reformats the text the layout created by a box move may be spoilt. To prevent this add the line '>FX ON' before the section with the box and the line '>FX OFF' after. This will disable reformatting in that section of text.

11. Rulers Tabs and Margins

A ruler is a special line that marks out the margins and tabs. Margins are the left and right hand limits of the text, whilst tabs are markers used to position text in set columns on the screen and are particularly useful for lining up columns of words or numbers. Tabs work in a similar fashion to those of a typewriter, but are much more flexible. The line at the top of the screen, immediately underneath the two status lines, is the 'active ruler line'. When Protext is initially loaded and there is no text present, this will be the 'default' ruler which is provided by Protext when no other default ruler is found. Each '!', '.' or 'C' on the ruler marks a tab position and the 'L' and 'R' indicate the left and right margins.

Ruler lines

Initially, with no document in memory, the default ruler supplied by Protext is displayed as the active ruler. If a document is loaded which contains any ruler lines, then the default ruler for that document is considered to be the ruler line in the document which will be in operation **when printing commences.** This is not necessarily the first ruler in the text, but the last ruler before the start of the text that will be printed. This ruler line will be considered to be the 'default ruler' for the duration of the document, or until such time as it is replaced by another one at the start of the printed text. The reason for this is explained later in the chapter.

At all times the 'active ruler line' at the top of the screen is the one applicable to the line that the cursor is on.

New ruler lines can be created to suit any required layout. The new ruler line is typed in as a normal line of text. The line must have a 'greater than symbol' or 'chevron' (>) in the first column. There is no need to type in the hyphens (-) along the line, though it may be preferred to make the ruler line more readily visible. Wherever a tab is required, a '!' should be typed in and the point '.' wherever a decimal tab is wanted. Columns 2 and 3 must contain either spaces, a hyphen, the letter 'L' or a '!' tab marker.

Note: The TAB key should not be used in a ruler line, when altering or creating a new ruler line since this will cause tabs to be set wrongly.

Any number of ruler lines are allowed in the text and each applies from the line beneath, until the next ruler line (or the end of the text if there are no more ruler lines). When the cursor is moved past a ruler line, in either direction, the first ruler line above the cursor will immediately replace the previously active ruler at the top of the screen. If the cursor is moved above the first ruler line in the text, the Protext default ruler will be restored.

Ruler lines allow the formatting of text in any chosen way. Simply altering the ruler line and using the CTRL-F command will reformat the text (alternatively FORMAT in Command mode will automatically reformat the entire document). If full use has been made of tabulating the text by inserting tab characters, moving the tab markers on the ruler line will cause the text to be repositioned immediately, using the altered ruler line. Thus it is easy to move columns of text/figures about or to experiment with different formats to decide which is best.

There are three editing commands connected with the use of rulers. The first of these is CTRL-D, which can be used at any time to copy the default ruler to the current cursor position. This can be particularly useful to restore the text to a standard format after using a special ruler to create tabulated columns of figures, for example.

The second command is CTRL-R which will restore **the previous ruler but one**. The most common use for this command is to restore the previous text format after, say, insetting the left and right margins of a piece of text to highlight it. One of the most common uses of ruler lines is to alternate between two different layouts and using CTRL-R at the end of each one will automatically copy down the ruler appropriate to the previous layout. In this way it is particularly easy to switch between two layouts simply by the use of CTRL-R.

A useful tip is to create an alternative ruler line at the start of the document, before any text is printed and before the first ruler line which will be used in the document. When the document is loaded, the second ruler line will become the default ruler (being the ruler line that applies to the first line of text). When the alternate ruler is required, CTRL-R can be used to copy it down and subsequently to alternate between the default ruler and the alternative one.

The third command connected with rulers is CTRL-V R. This command has the effect of 'hiding the active ruler line' at the top of the screen, and releases an extra line for displaying text. Repeated use of this command will toggle the ruler line on and off. It only has any effect on the ruler line beneath the status lines and does not hide the ruler lines in the document and even when it is invisible, it still functions as normal.

Tabs

The main use of tabs is for aligning text or numbers in columns. Protext has three types of tabs, the normal 'tab', the 'decimal tab' and the 'centre tab'. The decimal tab also serves the purpose of a 'right justification' tab. These tabs are inserted into a ruler line by typing in an exclamation mark (!) for a normal tab marker, or a point (.) for a decimal tab marker, wherever they are required. As many of either tab as necessary may be used on each ruler line.

When TAB is pressed (in insert mode) a special tab character is stored in the text. This character can be thought of as occupying several columns on a line, or as being one long, variable length space from the position where the tab was inserted up to the column marked by the tab marker in the ruler line.

Note: It is recommended that full use is made of tabs when creating a document, as not only do they provide a simple means of aligning text, but they enable the easy re-arrangement of text (See chapter on Formatting) and are most important when proportional printing is being used.

Normal Tabs

Each '!' on the ruler line indicates a tab position. Pressing the TAB key during text entry will move the cursor to the column appropriate to the next of these positions.

If insert mode is in operation, pressing TAB will result in a tab marker being inserted into the text, giving the appearance that a number of spaces have been inserted in the line. Pressing ←DEL once, immediately after TAB, will remove the tab marker and all the apparent spaces and move the cursor back to its previous position. Since the cursor cannot be moved within the columns formed by this 'long' space, it will jump over a tab character if moved onto it.

Pressing SHIFT-TAB will move the cursor to the next tab position, but will not insert a tab character. If overwrite mode is selected, the functions of TAB and SHIFT-TAB are reversed.

Decimal (right justification) Tabs

This is a special form of tab which can be used in a number of ways. It permits the alignment of numbers so that the decimal points line up automatically. Any numbers entered after pressing TAB to move to a decimal tab will automatically be positioned to the left of the tab position until the decimal point (.) is entered, after which any further numbers will be placed to the right. If no decimal point is entered (as for a whole number) and TAB or RETURN is pressed, the number will be right justified in such a way that units, tens, hundreds etc. all line up.

This tab can also be used with words, as a right justification tab, in which case the words will be lined up in such a way that the end of the words are all in line. This can be very useful for entry of words like 'Total' and 'Subtotal' in conjunction with columns of figures.

Note: Some countries use the comma instead of the full stop as the decimal point. Protext may be configured to accept the comma ',' instead of the full stop '.' by using the CONFIG utility but it should be noted that the full stop should still be used on the ruler line to mark the position of the decimal tab.

The right margin as a decimal or right justification tab.

The right margin (denoted by 'R' on the ruler line) also acts as a decimal/right justify tab. The TAB key should be pressed a sufficient number of times to move the cursor to the right hand margin of the document. Any text typed now will be right-justified at the margin. For example the address on a letter, or the date, may be usefully entered in this way.

Note: There must be sufficient space between the last tab and the right margin for the text that is to be entered, otherwise part of the text will be forced onto a new line.

Centre Tabs

A centre tab is marked by the letter 'C' in the ruler line. Text that is entered after pressing TAB to move to a centre tab is centred about the position of the 'C'. The effect of putting a single centre tab in the middle of the ruler line is the same as using the '>CE' stored command. This is particularly useful for centring text in a header or footer, especially proportional text.

Indent Tabs

The indent tab allows a paragraph to be indented an extra tab level without using a new ruler line with a different left margin. Pressing ALT-TAB at the start of a paragraph inserts an indent tab. On subsequent lines, until the end of the paragraph, text will wrap around to the position one tab in from the left margin.

If ALT-TAB is pressed more than once, the paragraph will be indented by one further tab position for each time that ALT-TAB is pressed.

Paragraphs may be indented before or after the text has been typed. To remove a level of indenting place the cursor in column 1 of the first line of a paragraph and press DEL.

The use of indent tabs greatly reduces the need for ruler lines in the text.

Ruler editing with the mouse

The active ruler line may be edited by using the mouse as follows:

Double clicking at a position on the ruler line where there is no tab position causes an ordinary tab to be set. Double clicking on a tab changes it to a decimal tab. Double clicking on a decimal tab changes it to a centre tab. Double clicking on a centre tab removes it. This works both on the status line ruler and the ruler in the text.

The left and right margins can be changed by dragging them in either direction. Move the mouse pointer onto the 'L' or 'R', press and hold down the left button, move the pointer until the 'L' or 'R' is in the required position and release the button. This works on the **status line ruler only**.

This method of editing the ruler will change the ruler that is currently active. If this is a ruler defined earlier in the text, then that ruler will be changed. If it is the default ruler, or the ruler in the text has been written out to disc, a new ruler line will be inserted into the text before the current paragraph. To create a new ruler press CTRL-D or CTRL-R first to insert a new ruler line.

Margins

The ruler line is also used to define margins. These are indicated by 'L' for left margin, and 'R' for right margin. These define the part of the screen within which text will be formatted. By default the left margin will be at column 1 and the right margin at column 70, but they can be set, in the same way as tabs, at any position (of course the right margin must be to the right of the left margin!).

To set margins a ruler line should be created, as described above, the cursor moved to the column at which the left margin is to be set and the letter 'L' typed. Next the position of the right margin should be selected with the cursor and the letter 'R' typed. If the left margin is to remain at column 1 then the 'L' should be omitted.

To see the effect of changing margins try creating a new ruler line above a paragraph of text with, for example, a left margin at column 5 and a right margin at 60. The status lines show the current column number and serve as a guide to selecting the correct columns. Positioning the cursor in column 1 of the first line of the paragraph and reformatting it with CTRL-F will rearrange the text to fit within the new margins.

Left margins and tabs - applications

1. Indentation of the first line of a paragraph

This is best done by setting a tab marker on the ruler line in the column to which the text is to be indented. At the start of each paragraph, pressing TAB, prior to typing the text, will position the cursor accordingly.

It is easy to indent the text at any subsequent stage, simply by moving the cursor to the start of the paragraph, pressing TAB, and then CTRL-F to reformat the paragraph. This is particularly useful for splitting a paragraph in two.

Example: a ruler line to give 57 column wide text with a tab at column 5 for indenting the first line of the paragraph.

```
>---!---------------------------------------------------R
```

2. Indentation of a paragraph or section of text

Indenting a paragraph is best done using the indent tab, as described above. At the end of the paragraph pressing RETURN will move the cursor to the left margin. In order to continue the indentation a new ruler line may be defined with the left margin at the required position. When RETURN (or SHIFT-RETURN or CTRL-RETURN) is pressed the cursor moves to the left margin, so, by setting an indented left margin, text can be automatically indented.

Note: A left margin should not be used to indent the whole of a document. This should be accomplished by use of the '>SM' side margin stored command (See Stored commands).

3. Marginal comments

When the text is formatted with the 'FORMAT' command any text that is within the left margin i.e. to the left of the left margin setting, at the start of a paragraph (the first line), is unaffected. If the formatting command, CTRL-F is used from edit mode, in order to reformat a paragraph, the effect will be exactly the same, unless the cursor is positioned in column 1 at the start of a paragraph when the command is used, in which case the whole paragraph will be moved within the left and right margins.

The exception is when indent tabs are used and this can be used to include marginal comments with the text at the start of a paragraph. Type the comment starting in column one and press ALT-TAB to move to the tab position required for the paragraph text.

For example the following ruler line may be found useful for numbering indented paragraphs:

```
>---!---!-----------------------------------------------R


    1.  First paragraph.

    2.  Second paragraph.
```

Another use for this type of layout is for the creation of scripts and one of the advantages is that marginal comments can be added at any time without disturbing the main body of the text, as long as they are restricted to the area to the left of the left margin. Care must be taken since formatting the text will leave text in the margin on the first line of paragraphs, but integrate comments on later lines with the main body of the text. It is preferable not to reformat paragraphs of this sort.

Note: A stored command, '>FX' is available and may be used to instruct Protext not to format specified parts of the document, either when the FORMAT command is used, or 'formatting during printing' is in operation. '>FX' is described in the chapter on stored commands.

Note: A command, FIXB, is also provided which enables the format of a section of text to be 'fixed' (See 'Commands'). Once the text has been fixed, it will not be affected by subsequent reformatting, but it will no longer be possible to edit it in the normal way, as all the soft spaces, tabs and returns which allow the reformatting of text are replaced with normal spaces and hard returns.

12. Formatting

Formatting is the process which determines the layout of the text and takes two forms, automatic formatting and manual re-formatting.

Automatic Formatting

There are two main features which determine the final appearance of the document and which happen automatically as the text is entered.

Word Wrap

It will already have been noticed that when a complete line of text is typed, the cursor moves on to the next line, and any part-entered words are also moved with it. This operation is called 'word-wrap'. The status line will indicate that word-wrap is working. CTRL-W toggles word-wrap on and off and the status line will change to indicate this. Setting word-wrap to 'off' and typing a line of text will show the difference. The cursor stays on the same line until RETURN is pressed. When the edge of the screen is reached, the display will scroll sideways. Pressing CTRL-W a further time will turn word wrap back on again.

Right Justification

It will probably also have been noticed that all text appears with a straight right hand edge, like a book. This is the default setting and the status line will indicate that right-justification is in operation.

Protext lines up the right hand edge by automatically spacing out the words on a line. CTRL-J may be used to turn off the right-justification, and the status line will change to indicate this. If a paragraph is now typed in, it will be found that the right hand edge is 'ragged', as if typed on a typewriter.

Note: The default settings of both Word-Wrap and Right-Justification may be altered to suit the user, by use of the CONFIG program.

Manual Re-formatting

There are two ways to force the text to be re-formatted. These commands will not normally be needed because Protext features automatic re-formatting of the text in most situations. That is, if a paragraph is edited, the correct format will be restored without the need for these commands. Sometimes it may be desirable to switch auto-reformatting off temporarily. This can be done by pressing SHIFT-CTRL-R.

Edit Mode Formatting

If the format of just a small part of the text has been destroyed, it is easier to re-format these small areas whilst still in edit mode. All that is required is to position the cursor on the first line that needs re-formatting and type CTRL-F, which will cause the remainder of the paragraph to be re-formatted according to the justification, word-wrap and ruler line settings. An alternative command is SHIFT-CTRL-F which formats the current paragraph without moving the cursor.

Command Mode Formatting

If considerable areas have been disrupted, or it has been decided to alter the number of characters on a line (see chapter on Tabs and Margins), the simplest way to re-format is to return to command mode, using ESC and then type in the command FORMAT. This will automatically re-format the entire text, from start to finish, in one go.

A variation of this command is FORMATB, which will format the part of the document selected with the Block markers (see 'Cut and Paste Editing' for details of block markers), but will continue past the end of block marker to the end of the paragraph.

Note: There are two commands which can be made to override the re-formatting of part or all of a document. All or part of the text may be 'fixed' with the command FIX (and FIXB to fix only part of the text). Details are given in the 'Commands' chapter.

The second method of overriding the FORMAT command is to use the stored command, '>FX ON/OFF'. This may be inserted into the document to turn formatting on and off, for sections of the text, as required. '>FX' is described fully in the chapter on stored commands. This is an example of a 'stored command' - a command that is put into the text to be used at some later time (In this case, during the course of formatting).

Centring text

It can be very useful to be able to position text, such as a title, exactly centred on the line. This can be done by moving the cursor to the line in question and pressing CTRL-C. The text will be centred and the cursor will be moved to the left margin of the next line.

If the file is re-formatted with a different text width (as explained in the previous section), the line will no longer be centred. To ensure that the line is always correctly centred, the three characters '>CE' should be typed in the first three columns of the line. This is another example of a 'stored command'. Protext will correctly centre enlarged titles when the characters are twice the width of the normal text. It is essential that either '>CE' or a centre tab is used if this is required.

Note: When proportional printing is in use, the '>CE' stored command or a centre tab should always be used to centre text. To centre text in a header or footer a centre tab is required (see 'Rulers Tabs and Margins').

Formatting whilst printing

Protext also has a stored command, '>FP ON/OFF' which may be embedded into the text and which ensures that the document is reformatted during the course of printing. See 'Stored commands' for full details.

Note: When mail merging is in use, any paragraphs containing merged in text will automatically be reformatted.

Soft spaces, returns and hyphens

Soft characters are used by Protext for formatting. When a paragraph is right-justified, spaces are inserted in the line. These spaces are treated differently from spaces inserted by pressing the space bar (these are called hard spaces). Hard spaces can never be removed by formatting, but soft spaces can.

There is a similar distinction between a hard and soft return (end of line). A hard return occurs where the RETURN key has been pressed, and this marks the end of the paragraph. A soft return occurs where the action of word-wrap has caused a new line to be started. Like soft spaces, soft returns can be removed by formatting.

A soft hyphen is slightly different in that it must be explicitly entered by pressing CTRL and hyphen and will be displayed in inverse. Soft hyphens can be inserted at points in a word where hyphenation is permissible. The formatter will then split a word at a soft hyphen and display a hyphen instead of moving the whole word onto the next line.

A soft hyphen will only be printed on the printer if it is at the end of a line, i.e. where the formatter has split the word. This feature is especially useful if there is a long word that is just too long to fit on the end of the line, but which might subsequently be moved by reformatting of the text.

Non-break spaces and hyphens

When the text is formatted it can result in there being several spaces where only one was typed. This is sometimes undesirable and Protext provides 'non-break spaces' which are not affected by formatting. A non-break space is entered by pressing CTRL-N, followed by the space bar and will be displayed as a small mark, so it can be distinguished from a normal space. Soft spaces will not be inserted between words linked by the non-break space and the words will not be separated at the end of a line.

Examples of use:

1. The space between 'Mr' and 'A' in 'Mr A.Smith'. This would look wrong if displayed as 'Mr A.Smith', so CTRL-N space is used after 'Mr' instead of just pressing the space bar.
2. A string enclosed in quotes, that must be printed exactly as it was typed, e.g. 'T h i s s e n t e n c e i s s p a c e d o u t'.

Non-break hyphens are similar. Normally when a hyphen is used in a word, if that word comes at the end of a line, the word will be broken at the hyphen and the second part of the word printed at the start of the following line, as would be expected. Sometimes this is not the effect that is required and the two words should be kept together as one. An example of this is found many times in this manual, where editing commands are described. A command like 'CTRL-F' would normally be split onto two lines if it came at the end of a line, which would look wrong. Protext provides a non-break hyphen to take care of this situation. Instead of typing a normal hyphen,. typing CTRL-N, followed by pressing the hyphen key will set a non-break hyphen instead. This will be displayed as a hyphen on a vertical line, but when printed, will appear as a normal hyphen.

13. Find and Replace

Two commands, FIND and REPLACE are provided, which permit searching through text for any string of characters and, if specified, replacing them with a second string.

Typing FIND or REPLACE from command mode will result in a request for the 'FIND string', followed by a request for the replacement text (if the REPLACE option was selected). After entering the string (or strings) of text, one or more of a number of options may be selected by typing the appropriate letters one after another (in any order). Each option is either a single letter abbreviation or a number (these are listed on the screen). Pressing RETURN on its own will cause no options to be selected.

If the 'Find text' or 'Replace text' option is used from the menu a dialogue box will appear which gives one or two text boxes in which the strings should be typed, and an option button for each of the available options. When the required options have been selected, click on the 'GO' box to perform the search.

Options available

The options available are as follows:

G	Global search. If selected the whole text is searched from the start, otherwise only the text from the current cursor position to the end of the text.
C	Case specific search. If selected all letters will only match letters that are the same case, otherwise either capitals or lower case letters will be treated as being the same.
W	Find string only if it appears as a complete word. For example to find occurrences of the word 'and' without finding 'hand', 'England' etc.
B	Search backwards. Searches from the cursor position to the start of the document, or from the end if G is also selected.
A	Find or replace all strings automatically. REPLACE will change all occurrences of the string with the new one, without requesting confirmation and return a figure of the total number of replacements made. In the case of FIND being used, it will simply return the total number of occurrences of the string.
S	Ignore any spaces in the text while attempting to match the search string.

n Find or replace the nth occurrence. n should be a number between 1 and 255. This option has a number of uses, but a simple example might be to check that every set of quotation marks has a matching closing set, in which case FIND would be used to find '"' and '2G' would be specified as options, to search globally for every second occurrence.

If no options are selected the search will be forwards, from the current cursor position to the first occurrence of the string, ignoring the case of letters, finding the string even if it occurs as part of a longer word, and asking for confirmation before replacing a string.

Any number of wildcards are allowed in the string. A wildcard is a character that matches any character in the text, except the return character. It is entered in the string by typing a question mark (?).

A tab character may be entered simply by pressing the TAB key. It is displayed as a right pointing triangle.

There are various characters that cannot be entered directly, but that it may be useful to include in a search string. These include return characters and printer control codes. Provision has been made for including these in a string, by means of an 'escape character'. The escape character (!) should be typed in, followed by a symbol, number or letter, as appropriate. Thus to search for a printer control code, type '!', followed by the code letter.

The full list of characters that are entered by this means is:

printer control code	!<letter>
question mark	!?
exclamation mark	!!
single quote	!'
double quote	!"
hard return	!.
soft hyphen	!-
spell ignore marker	!=
index word marker	!*
index phrase marker	!#
footnote marker	!~
non-break hyphen	!<underline>
non-break space	!<space>
search for specified code	!<number>

Using FIND

Once the string and any options have been selected, edit mode is entered and the cursor placed on the first character of the first occurrence of the string. To find the next occurrence of the string, the f6 key should be pressed. This need not be done immediately. Editing can be carried out first and when complete, the search may be continued by pressing f6. At any stage, f5 can be used to search back towards the beginning, if necessary.

As with other commands, FIND can be used by typing the string on the same line as the command name, followed by any options. Thus the command 'FIND word GWC' will search for the string 'word' from the start of the document, selecting only those occurrences where it is a complete word with all letters in the same case as specified. If no options are specified, the default options will be used.

If the A option is selected, Protext will return the total number of occurrences found when the search is complete.

Using REPLACE

The cursor will be positioned on the first character of the string and a dialogue box will be displayed giving the options 'Replace and continue', 'Skip and continue', 'Replace and stop' and 'Cancel'. Either click on the appropriate box or press the indicated key. Pressing 'Y' will replace the string with the new one and the cursor will move to the next occurrence. Pressing 'N' will leave the string untouched and move the cursor to the next occurrence. Pressing 'X' (exit) will replace the string and return to edit mode. Alternatively ESC may be pressed and normal editing resumed without changing the string. At a later time, f6 may be pressed to resume the find and replace operation. Alternatively f5 may be used to resume the search in the reverse direction, which may be found useful if an occurrence of the string is passed over by pressing 'N' in error.

If option A is selected then all occurrences of the string are replaced without prompting and the program will remain in command mode. When complete a count of the total number of changes made will be displayed.

When replacing a string the case of the string being replaced is preserved unless the find and replace string were typed in different cases (see examples 6 and 7 below).

Wildcards in replace strings

The wildcard character '?' can be used in replace strings. There must be at least as many wildcards in the find string as in the replace string. The meaning is 'leave the corresponding wildcard character unaltered'. This can be extremely useful. As one example, suppose a list of code numbers of the form 'NEW12345' has to be changed so that the numbers are all in the form '12345/OLD'. The command to do this is:

```
REPLACE NEW????? ?????/OLD
```

Find and replace within block

FINDB (FB) and REPLACEB (RB) can be used to restrict the range of the search to the currently marked block. All the options described above may be used. The 'G' option will often be used so that the whole block is searched.

Find word at cursor

It is often useful, whilst editing, to move directly to the next occurrence of the word on screen at the cursor position. This can be done either from the Text menu or by the following keys:

SHIFT-CTRL-f5	move back to previous occurrence of word
SHIFT-CTRL-f6	move forward to next occurrence

Examples

1. To find all occurrences of the word 'text' in lower case only, starting at the cursor position.

```
FIND string: text
Options: CW
```

2. To convert all occurrences of 'rom' or 'Rom' to 'ROM', asking for confirmation of each replacement.

```
FIND string: rom
REPLACE with: ROM
Options: GW
```

3 To find the 12th 8 letter word.

```
FIND string: ????????
Options: WG12
```

4. To replace bold codes, changing the on and off codes to different typesetting commands. Note that quotes are needed in the first command as the comma would otherwise mark the end of the replace string.

```
REPLACE !b '<nr5><ps10,12>' AG2
REPLACE !b <nr6><ps12> AG
```

5. To match '1,2' '1, 2' and '1 , 2'

```
FIND "1,2" S
```

6. To force all occurrences of 'Protext' and 'protext' to upper case.

```
REPLACE Protext PROTEXT G
```

7. To change all occurrences of 'Croydon' to 'Peterborough' and 'CROYDON' to 'PETERBOROUGH' (preserving the original case).

```
REPLACE croydon peterborough AG
```

14. Languages and Special Symbols

Protext is capable of being used with most European languages and supports the use of accents and special characters.

Characters containing accents may be typed in during the course of editing and will appear correctly on screen. They may also be stored in the dictionary and will be checked and may be corrected in the normal way.

The base character should be entered first and then immediately followed by ALT and the appropriate accent key (see below for details of keys). The accent will then be positioned over the character. The keys have been selected for their visual similarity to the accent they produce.

If an accent is required by itself, press space followed by the accent key. Should any of the accent characters be required frequently it is possible to redefine the keys to give just the accent, or to put the accented character in a macro on a suitable key.

Accents

Key to press		Accent
ALT-'	´	acute
SHIFT-ALT-~	˘	breve *(Amiga: use menu)*
SHIFT-ALT-^	ˇ	caron (háček)
ALT-^	ˆ	circumflex
ALT-"	¨	diaeresis/umlaut
SHIFT-ALT-'	˝	double acute *(Amiga: use menu)*
ALT-\	`	grave
ALT-_	¯	macron
SHIFT-ALT-.	˙	overdot
ALT-%	˚	ring
ALT-~	˜	tilde
SHIFT-ALT-,	¸	cedilla
ALT-;	˛	ogonek

Special characters

In addition, the following keys are pre-defined to produce:-

Key to press		Character
ALT-C	ç Ç	c cedilla
ALT-E	æ Æ	ae diphthong
ALT-L	ł Ł	l slash
ALT-N	ñ Ñ	n tilde
ALT-O	ø Ø	o slash

The above characters can be entered in upper case by pressing SHIFT and ALT.

PC Note: In text mode ø and Ø are displayed as ¢ and ¥ on the screen but print correctly.

ALT-H	½	Half
ALT-Q	¼	Quarter
ALT-S	ß	Double s (German)
ALT-←	←	Left arrow
ALT-→	→	Right arrow
ALT-↑	↑	Up arrow
ALT-↓	↓	Down arrow
ALT-<	«	Opening quotes (French).
ALT->	»	Closing quotes (French).
ALT-?	¿	Inverted question mark
ALT-!	¡	Inverted exclamation mark

Note: The ALT-<letter> combinations are defined as macros, so care should be taken when selecting keys for macros, if any of these characters are required. An explanation of the way to remove existing macro definitions is given in the 'Hints and Tips' chapter.

The choose character window

The 'choose character' option in the Text menu allows any of the available characters to be entered into the text. A window is displayed containing all the available characters which may be selected by moving the cursor keys and pressing RETURN or by clicking on the required character. This method is recommended for characters that are used only occasionally.

Changing the keyboard layout

Protext can configure the keyboard to use a number of different layouts for different languages. This will normally be done using CONFIG when Protext is first installed, but it can sometimes be useful to temporarily change layout whilst editing. This can be done by pressing SHIFT, ALT and a function key. The function keys select languages as follows:

f1	U.S. English	f6	Italian
f2	U.K. English	f7	Swedish
f3	French	f8	Danish
f4	German	f9	Norwegian
f5	Spanish	f10	Portuguese

Greek letters *PC / ST only*

PC / ST *Certain Greek letters are provided in the character set and may be printed on a printer supporting the IBM character set. An exec file is provided containing macro definitions which assign these Greek letters to keys as listed below. To use the Greek letters ensure the file 'GREEK' is available in the current directory or on the path and type the following command:*

EXEC GREEK

Greek characters will be available as follows:

ALT-A	α	*alpha*			
ALT-B	ß	*beta*			
ALT-D	δ	*delta*			
ALT-E	∈	*epsilon*			
ALT-F	ϕ	*phi*	*SHIFT-ALT-F*	Φ	*upper case phi*
ALT-G	Γ	*gamma*			
ALT-M	µ	*mu*			
ALT-P	π	*pi*			
ALT-S	σ	*sigma*	*SHIFT-ALT-S*	Σ	*upper case sigma*
ALT-T	τ	*tau*			
ALT-V	Ω	*omega*			

PC *Certain keys on the numeric pad have also been configured to produce some of the IBM graphics characters (IBM PC version only). SHIFT-CTRL-Keypad1 produces a 'lower left corner', SHIFT-CTRL-Keypad2, a horizontal line, SHIFT-CTRL-Keypad3, a 'lower right corner' and so on, using the number keys round the outside of the pad. SHIFT-CTRL-Keypad4 and SHIFT-CTRL-Keypad5 are used for other functions so SHIFT-CTRL-Keypad8 gives a vertical line.*

Characters not normally available may be entered using ALT and the keypad numbers. This is similar to the facility available under MS-DOS with the difference that a leading zero must be typed. For example the Greek letter alpha may be entered by typing:

ALT-0224 *(hold ALT while typing the numbers)*

The code numbers for other characters will be listed in the MS-DOS manual.

Note: If any of the other graphics shapes are required, they may be entered into the text using the choose character window (see above).

Line drawing

A much easier way of using the line graphics is provided by 'line drawing mode'. Corner symbols are inserted automatically as well as the appropriate characters when lines meet or cross. There are two commands to engage line drawing mode:

CTRL-f7 turns on line drawing using the line graphics.
CTRL-f8 asks for a character to be typed and then turns on line drawing mode. In this case the lines will be drawn simply using the chosen character.

To draw lines hold down ALT-→, ALT-↑, ALT-← or ALT-↓ and a line will be drawn in the chosen direction. Either CTRL-f7 or CTRL-f8 will subsequently turn line drawing mode off. To erase a line previously drawn use CTRL-f8 and press the space bar. Then retrace the line with ALT and the cursor keys.

Note: Printer requirements

Not all printers will be able to print all the accents or line graphics. See the note under 'Equipment requirements' in the Introduction chapter.

15. Miscellaneous Editing Commands

Timed automatic save

When this option is chosen in CONFIG, Protext will, after each specified time interval, if the file has been changed, save a copy of the file. If the file size is larger than a value specified with CONFIG (default 10K) a dialogue box gives the option to save the document. If no response is given to this dialogue box within 30 seconds, the document will be saved anyway - thus protecting the data if the computer is left with the file unsaved.

When a file is automatically saved, it is given the extension '.SAV'. For example if the file being edited is called 'REPORT' and the file is lost (for example due to a power failure), the file can be recovered by loading 'REPORT.SAV' and saving this as 'REPORT'.

Add column or row of figures

The addition function is used from the Text menu. Selecting the 'Addition mode' option puts Protext into addition mode. The word 'Total' will be displayed on the status line to indicate this, together with the running total.

At this point the calculated sum is reset to zero. To add numbers that are on the screen simply move the cursor over the numbers. Thus a column can be added by pressing ↓ or ↑ to move the cursor over the column, or a row by pressing → or ←. Alternatively click on the required numbers with the mouse and Protext will add all numbers visited.

When all the required numbers have been added use the menu option 'End addition' to cancel addition mode. To insert the result in the text place the cursor at the required position and use the menu option 'Insert addition result'. This also terminates the addition so if the cursor is in the correct place there is no need to use 'End addition' as well. The result is inserted showing the maximum number of decimal places used in the data.

Date and time

Keys are defined to insert the current date or time into a document.

ALT-D	Inserts date into text
ALT-T	Inserts time into text

Viewing the text without control codes

Control codes of various kinds (printer control codes, place markers etc) are normally displayed in the text in inverse video. This means that the formatting of the text (tabulated columns of figures, right-justification, etc.) may be unclear. Pressing f4 or selecting 'View codes' from the menu changes the display of printer control codes as well as other special markers such as index markers. Displaying all codes is most useful when it is required to make changes to those codes that have been entered. This option shows exactly what codes have been stored in the text and in what order. The f4 key is therefore a very important function. The control codes affected are as follows:

Printer control codes (a-z), block markers ([,]), place markers (0-9 and ?), special markers (=, * and #) non-break spaces, and soft hyphens (-).

The effect of pressing f4 depends on which of three view states Protext is using at the time.

The first state is 'all control codes hidden' or full wysiwyg. Pressing f4 then takes you to the second state which is 'all control codes visible'. In this second state attributes such as underlining are not shown, except by the control code. Pressing f4 returns to the first state.

The third state may not be entered by pressing f4 but occurs when a control code other than one of those attributes shown on screen is entered, for example 'condensed'. The control code just entered is displayed, as are codes for all attributes not shown on screen.

Similarly when an index marker is entered just the index and spell ignore markers are displayed. When a place marker is entered just the other place markers are shown. Pressing f4 from these states hides all control codes and subsequently switches between showing all or no codes.

Protext will display certain characteristics on the screen ('wysiwyg' style). The characteristics displayed and the method depends on the system being used.

PC	*If a colour monitor is being used (CGA, EGA, MCGA or VGA), bold, underlining and italics will be shown by using different colours. The colours may be configured using CONFIG. If a monochrome (Hercules) adaptor is being used, underlining and bold are shown naturally.* *Additionally, when graphics mode is being used, bold, underlining and italics are shown as such. VGA and EGA usually show different colours for these attributes as well. Subscript and superscript are shown on VGA, MCGA and EGA.*
ST	*The following attributes are shown on screen whilst editing: bold, underlining, italics, subscript, superscript.*
Amiga	*The following attributes are shown on screen whilst editing: bold, underlining, italics.*

Viewing the text with tabs, hard returns and hard spaces

The command CTRL-V T toggles the display of tab characters and hard returns on and off. A tab is shown as a right pointing triangle and a hard return as a cranked left pointing arrow. A second use of this command will turn the display off again.

The command CTRL-V S will display all hard spaces (those that are typed in at the keyboard) as a shaded pattern on the screen. Pressing CTRL-V S a second time will restore the display to normal.

These options are primarily intended for the purpose of viewing where the various characters have been placed, particularly when sorting out complicated layouts and by default they are in the 'OFF' setting when Protext is loaded. Some people prefer to have some or all of these options in operation at all times and the CONFIG program provides the option to adjust the defaults to suit the user.

Page break lines

The command CTRL-V P will turn on and off the display of page break lines. Page breaks are shown by a thick line extending over the full width of the screen.

Auto indent

This works in program mode only.

This feature is particularly useful when editing programs written in a block structured language such as C or Modula 2. When auto indent is enabled the effect of the RETURN key is modified. Instead of moving the cursor to column 1 the cursor is moved to the column that contains the first non-blank character on the line above. Thus it becomes much easier to type indented blocks of code.

To turn auto indent on or off, type SHIFT-CTRL-I. Alternatively CONFIG may be used to select auto indent as the default.

When RETURN is pressed, tabs are inserted at the start of the line. The level of indenting can easily be changed.
Press RETURN and TAB to indent a further level.
Press RETURN and ←DEL to come out one level.

43 and 50 line displays *PC EGA/VGA/MCGA/Hercules and ST mono only*

PC/ST	*Pressing ALT-F whilst in edit mode will switch between the normal 25 line display and a display with more lines. On an EGA or Hercules monitor there will be 43 lines, on a VGA there will typically be 50 lines. On an ST in high resolution mode there will be 50 lines. Pressing ALT-F again will restore the 25 line display. Note that ALT-F is defined as a macro and may be redefined.*

Graphics and text mode

PC	*Pressing ALT-G whilst in edit mode will switch between text and graphics mode.*

Window resizing *Amiga only*

Amiga *Protext operates within an Amiga window and is compatible with the Amiga Intuition standard. This manual describes commands in the form that is common to all versions of Protext. On the Amiga, though, most commands are available from pull down menus which will appear when the right mouse button is depressed.*

The width and height of the window may be changed at any time when in edit mode by clicking on the 'resize' icon at the bottom right of the screen and 'dragging' the window until it is the required size. The window can be repositioned (if it occupies less then the full screen) by clicking on the menu bar and dragging. In command mode the sizes may only be increased - to decrease the window size press ESC to enter edit mode.

Note: Resizing a window requires some memory to function, so if there is insufficient memory it may fail.

WB 2.0 *Under Workbench 2.0 the window also has a 'zoom gadget' at the top right of the window (to the left of the depth gadget). Clicking on the zoom gadget collapses the window to its minimum size, clicking again restores the previous size.*

16. Multiple File Editing

Protext provides the facility to work on several documents at the same time. These documents are maintained quite separately and are loaded and saved individually. Any operation can be carried out on one document without affecting the others, the cursor location and all markers being maintained separately for each document. Blocks of text can be copied between one document and another.

All of these functions are available from the menus and commands.

To load a new document, the swap menu should be used. This menu lists all the documents that are currently open for editing. If only one file has been loaded so far the menu will show that file and also an entry for 'New file'. Selecting 'New File' will switch out the current document and leave an empty document. The new document should be loaded in the normal way. The swap menu will now show the names of the two open files and 'New File'. To load a third document selecting 'New File' and proceed as before.

To switch between documents simply use the swap menu and select the required document. Switching to a new document will cause the information on the status lines to change to suit the current document, enabling easy recognition of which document is being worked on. If a colour display is being used the status line colour will change when documents are swapped.

Alternatively pressing ALT and a function key will swap to the document number corresponding to the function key, or the command SWAP can be used. This has various forms:

SWAP	same as CTRL-Y (see below)
SWAP n	swap to file number 'n' (same as ALT-Fn)
SWAP <filename>	swap to named file, or load it into an empty document if not already open
SWAP 0	swap to the first empty document (same as selecting 'New File' from the menu)

To show the list of currently open documents either look at the swap menu. Here a tick indicates that a file has been changed and not saved. Alternatively use the command 'SHOW' which indicates the same and also shows the file size and the original full pathname.

Copying a block between documents

At any time that two or more documents are open, two of them occupy a special position. One is the current file and the other is the alternate file, which is used for copying blocks.

In edit mode, CTRL-Y swaps between the current and alternate document, making the alternate the new current and vice versa.

When only two files are open the non-current file will always be the alternate file. To change the current file either load a new file or swap to an already open file using the menu. To change the alternate file do the same but press CTRL-Y before and after. By splitting the screen (see below), this process is made much easier as both the current and alternate files are always visible.

The CTRL-O (letter o) command is used to copy text from the alternate document to the current.

Before a block of text can be copied over, the block should be marked out using the markers in the normal fashion. Typing CTRL-Y will swap files and the cursor should be positioned where the text is required. If CTRL-O is then pressed, the block will be copied across at the current cursor position.

If the original text is no longer required, CTRL-Y should be pressed again, to return to the original document, followed by pressing CTRL-DEL, to delete the original text.

Boxes may also be copied between the two files by selecting box mode in the file containing the box to be copied.

To transfer text from one file to another - load the first file, type SWAP, load the second file and use CTRL-O to copy the blocks required into the first file, before re-saving it. This is quicker than using SB (save block), loading the other file and merging the saved block of text into the document and finally resaving it.

Split screen editing

The swap menu gives the option to split the screen so that two documents (the current and the alternate) may be viewed at the same time. The screen will be split horizontally across the middle. The number of lines allocated to each file can be altered by dragging the dividing bar up or down. Alternatively, if a mouse is not being used the 'WINDOW' command has the same function:

```
WINDOW <number>
```

The number is the number of lines allocated to the bottom file. Typing 'WINDOW' by itself splits the screen into equal parts, or unsplits the screen if it is already split.

When the screen is split, CTRL-Y still swaps from one document to the other. The flashing cursor will move from the top to the bottom or vice versa, and the status line colour will change.

The status line always shows the information for the file in the top window, while the dividing bar shows limited information about the bottom file. If full information is required while editing the bottom file, the two windows can be exchanged using the swap menu or by pressing SHIFT-CTRL-Y.

Linked files

When the screen is split into two windows the files can be linked so that a single command will scroll both windows. To link the files use the option in the Swap menu or the command 'LINK ON'. Then SHIFT-↑ and SHIFT-↓ will scroll both files at the same time. This is useful when the two files contain related data, such as when translating, as the information can be synchronised.

The Swap menu option or 'LINK OFF' cancels the file linking.

17. Printing

Printer drivers

Protext is supplied with a number of printer drivers to suit a wide range of printers and the drivers to be used will be selected as part of the installation process. The PROTEXT.CFG configuration file contains details of the default printer driver to be loaded at the same time as Protext.

A printer driver is a collection of codes that a particular printer uses, including the codes for underlining, bold, etc. With one exception (the 'simple printer driver') all printer drivers are stored as files with the suffix '.PPD' (Protext Printer Driver).

The simple printer driver is built in to the program and is designed to work with virtually all printers, since it uses no special codes, but just assumes that the printer is capable of backspacing. In the event that no printer driver has been loaded into Protext, this printer driver will be used. The simple printer driver only supports underlining and boldface print.

Note: It should be appreciated that many of the special printing effects that are possible on a dot matrix printer or laser printer may not be available on a daisywheel printer. It should also be noted that not all dot matrix printers have all the facilities mentioned in this chapter.

Note: It is important that Protext is configured to use the correct printer driver. See the 'Protext printer drivers' booklet.

When Protext is initially loaded, it will automatically load the default printer driver specified in the PROTEXT.CFG configuration file. If no printer driver is specified, or for any reason the specified driver is not found, Protext will automatically configure itself for a simple printer. The name of the driver that has been loaded will be displayed on the screen.

Different printer drivers may be loaded at any time which means that if both a daisy wheel and a dot matrix printer are connected to the computer, it is possible to print out a draft copy on the dot matrix printer and then do the top copy on the daisy wheel printer. Printer drivers may be loaded either from command mode using the PRINTER command (See 'Commands'), or from within the document by using the stored command '>PR' (See chapter on 'Stored commands').

What are printer control codes?

Printer control codes are used to control the special features of a printer, such as underlining, different sizes and styles of print, subscripts, superscripts and selecting different character sets.

Each make and model of printer has its own control codes and Protext needs to know what printer control codes the printer in use requires to carry out the different type styles and effects. This information is stored in the printer drivers.

Protext's printer control codes are special single characters, which are embedded into the text whilst editing. These are recognised as such by Protext when it is sending text to the printer, at which point it substitutes the sequence of numbers appropriate to the code and sends those to the printer instead. The result of this is that once a suitable driver has been created (if necessary), inserting a single code into the text can have the effect of sending a complete sequence of codes to the printer.

How to use printer control codes

Printer control codes may be entered anywhere in the text. This is done by typing CTRL-X followed by a letter, to define the code. Bold, italic and underline may also be entered more conveniently with ALT-B, ALT-I and ALT-U respectively. Any letter from 'a' to 'z' can be defined to represent any code, but as standard, several have been allocated uses covering the most common features:

@	printer reset code
b	bold (emphasised)
c	condensed print
e	elite (12 cpi)
i	italics
l	enlarged
n	normal (pica) (10 cpi)
p	proportional
q	near letter quality (NLQ)
s	subscript
t	superscript
u	underline

When one of these codes is entered, either the effect of the attribute is shown in the following text or the letter appears in the text in inverse video display. In most cases the first occurrence of the code turns on the appropriate feature and the second occurrence turns it off

For example: To enlarge a single word, the cursor should be moved to the start of the word and CTRL-X pressed, followed by L. The cursor should then be moved to the end of the word and the same sequence repeated. To underline a word do the same but press ALT-U instead.

Each printer code occupies one column on the screen, but Protext takes this into account and when the text is printed, justification will be correct. Any tabulation or formatting will appear visually wrong when printer codes are visible on the screen, so a command is included to overcome this problem and allow the text to be viewed without the printer codes. This is CTRL-V V or f4. Pressing f4 a second time restores the display of printer codes.

Character pitch

Pica and elite are the two most common sizes of fixed pitch printing. Typically the 'n' and 'e' codes that select these have no meaning for the 'off' code. They are mutually exclusive styles of print, so turning one off is done by selecting the other.

The proportional code 'p' is slightly different in that turning it on instructs the printer to use a proportional printing font. The operation of other codes in conjunction with 'p' depends on the printer being used. Proportional printing is covered in detail below.

Note: Some printers will only commence printing in the new pitch on the line following the one in which the control code was used.

In addition to the above pitches, it is possible on many printers to use further control codes to implement condensed or enlarged styles of printing and with most printers this has the effect of reducing or enlarging the typeface currently in use. In other words, 'condensed elite' and 'condensed pica' are both possible and may result in different size characters.

The following list gives an indication of typical values for the numbers of characters printed per inch when the appropriate control codes are specified. By default, Protext will print in draft quality with a pitch of 10 characters per inch.

		cpi
Pica. -	If pica is in use:	
n	pica	10
l	enlarged pica	5
c	condensed pica	17
lc	enlarged/condensed	8.5
Elite. -	If elite is in use:	
e	elite	12
l	enlarged elite	6
c	condensed elite	20
lc	enlarged/condensed	10

The Print dialogue box

When 'Print text' is selected from the Print menu the print dialogue box appears. This enables selection of the destination for the output (printer, screen or file) and gives the choice of various options which are discussed below. If output to file is selected a filename must be given in the text box provided.

The print dialogue box also appears when 'Print a file' or 'Print block' is selected. In the case of printing a file, the file selector is called up first so the file to be printed can be chosen, then the print dialogue box follows.

Printing commands

Protext provides many different printing commands, but for most purposes only one or two will be used. All the printing commands begin with the word 'PRINT', followed by further letters in many cases. These letters describe exactly what the commands do. See the 'Commands' chapter for full details on any of the commands mentioned below.

'PRINT' used on its own will print a document in draft mode, unless any printer control codes in the document specify otherwise.

Near letter quality (dot matrix printers)

Print commands with the letter 'Q' added, such as 'PRINTQ' and 'PRINTPQ' will automatically ensure that the document is printed in Near Letter Quality, without the need to use the printer control code in the document. This is particularly useful as it means that a draft copy can be printed first, using the simple command, then the 'top' copy produced with the same command, but with the 'Q' suffix. For example, PRINT, followed by PRINTQ. This assumes that the printer is capable of printing in NLQ mode and the codes are set correctly in the printer driver.

Printing by pages

A further group of printing commands have the 'P' suffix, such as 'PRINTP' and 'PRINTPQ'. These commands allow the selective printing of Pages and when used, confirmation is requested as to whether each page is to be printed.

These commands also allow a list of pages to be specified, and if this is done, just those pages will be printed without requesting confirmation. For example 'PPQ 3,9,11-13,84'.

Printing a block

A third group of printing commands use the letter 'B', as in 'PRINTB' or 'PRINTQB'. These commands will print only a Block of text marked with the block markers, rather than the whole document. As can be seen from the above, in many cases combinations of the extra letters are used, such as 'PRINTQB'.

Printing from disc

If a filename is typed after the command the file of that name will be printed from the disc, rather than the file in memory. For example, 'PRINT LETTER' will print the file called 'LETTER', if it is found in the current drive and directory.

Number of copies

The number of copies to be printed may be specified as a number. For example, 'PRINT 3' will print the current text 3 times and 'PRINT MYLETTER.DOC 5' will print 5 copies of the file called 'MYLETTER.DOC',

Print to screen

All the normal printing commands which send output to the printer have now been covered, but in addition there are two further groups of printing commands which send print output to other places. The 'PRINTS' and 'PRINTSB' commands send the output to the Screen instead of the printer. This provides a convenient method of checking on page breaks and the visual appearance of the document, before doing a proper print. It is also particularly useful for checking that a mail merging file is giving the expected results. The optional parameter of a filename can be used to print a disc file to the screen and this is useful as a means of checking the contents of a disc file.

Print to file

The 'PRINTF' and 'PRINTFB' commands function in a similar fashion to the PRINTS commands, except that instead of sending the output to the screen, it is sent to a disc as a file, complete with all printer control codes. In this case it is necessary to specify a filename with the command and this will be the name that the file will be saved with. Optionally, a second filename may be specified and in this case, a file of that name will be read from disc and sent to a file with the first filename specified.

What determines the form of the printed output?

Protext has a natural order in which it uses commands. If Protext is loaded with no printer driver specified, then the default 'simple printer' driver will be used. If a printer driver to suit the printer is loaded, then this will override the simple driver and the codes specified in the printer driver will be used when any printer control codes are inserted into the document. If any new or different printer control codes are specified within the document, using the '>CC', '>RC' or '>OC' stored commands, then these in turn will take precedence over the codes provided by the printer driver. See 'Stored commands'.

Similarly, if a document is created and contains no stored commands, then the default settings provided by Protext for such things as page length, line spacing etc. will be used. If Protext has been reconfigured to alter the default settings, using the CONFIG program, then when the PROTEXT.CFG file is loaded, the original default settings will be replaced by the new ones provided in the configuration file. These settings may, in turn, be modified by the use of stored commands in the text.

The appearance of a document is therefore determined by a number of factors. Starting at the very beginning, the appearance will be governed by the settings of rulers and tabs in the document and whether right justification is turned on or off when the text is entered. The appearance can be further altered by the use of stored commands, which may be used to change the default settings for the sizes of the various margins, the page length, whether single or double line spacing is to be used, headers or footers printed on each page and many other features. Each of these stored commands will override the provided or reconfigured default settings. In addition to these features, it is possible to use printer control codes to specify the type style which is to be used in different parts of the text and, assuming a suitable printer driver has been loaded, these will override the simple printer driver.

Background printing

All output to the printer passes through Protext's background printer buffer, and the printing is controlled by the commands described below. The only observable effect apart from reduced time spent waiting for the printer, should be when printing on single sheets. In this case, at the end of every page that is sent to the printer, a message will appear on the screen, as follows:

```
Press RETURN after inserting next sheet of paper or ESC to stop printing
```

This message replaces the 'Page n. Press SPACE to print' message, except when using the command PRINTP (PP).

Pressing ESC at this stage will put the printer on stop, with the same effect as the STOP (ST) command. The command CONT (CO) should be used to continue the printing for the next sheet. See 'Commands' for descriptions of the commands ABANDON, BACK, CONT and STOP.

Microspacing

This is designed to be used with characters of fixed width, for example, a fixed pitch daisywheel or a non-proportional font on a dot matrix or laser printer.

Microspacing is a means of simulating proportional printing to a degree. Microspacing evens out the spaces between the words on a line, and so gives a better appearance to the text. It does not adjust the widths of the characters and would normally only be used with a printer that was **not** capable of proportional spacing.

Microspacing will happen automatically on many printers. If using a printer driver which does not microspace by default the stored command '>MS ON' may be used to enable microspacing. About the only printers to which this applies are 9 pin dot matrix printers and on such printers microspacing tends to be very slow and so is not recommended. '>MS OFF' will turn microspacing off, so it is possible to print a document using microspacing on only part of the text.

Printing in program mode

The PRINT command operates differently in program mode. Stored commands are printed and not obeyed, and the print options are ignored. Instead, the text is printed continuously with no margins and single line spacing. This is suitable for printing program listings and mail merge templates or programs, and is also useful to print a copy of the text with the stored commands shown. In the latter case enter the PROG command before printing and the DOC command afterwards. Printer control codes and character redefinitions are both still effective.

Note: It should be noted that whilst any control codes in a program file will be acted on by the printer, it is not possible to insert them into the text whilst in program mode. If any control codes are required (e.g. to print the document in a more condensed typestyle), they may be inserted by going into Doc mode, entering them in the normal way and then reverting to Prog mode.

Newspaper column printing

Protext is capable of formatting and printing the text in up to 26 columns of chosen width and spacing between columns. This is sometimes known as 'snaking'. To do this the stored command '>PC' (print columns) is used to specify the number of columns. This should be followed by a ruler line whose margins correspond to the required width of the column of text.

The columns will then be printed evenly spaced within the width of the ruler line active at the point that '>PC' is used.

```
>PC 3        ; 3 columns spaced evenly within the normal text width
>PC 1        ; reverts to normal, single column printing
```

A further option allows the total width of the text to be changed and not taken from the ruler line. A second parameter is used with '>PC' which is the width of the text in characters, equivalent to the column number containing the right margin of a ruler.

```
>PC 2 75 ; will print the text in 2 columns with total width 75 characters
```

When editing, the text will appear on screen as one long column. The line counter on the status line will indicate which column the current line will be printed in by means of a letter. For example the first line on the page will be shown as 'A1', the second line in the third column as 'C2', and so on.

One further stored command may be used when it is necessary to terminate a column before the end of the page. This is '>EC' (end column) and the effect is similar to a page break, except that printing resumes at the start of the next column on the same page.

The effect of the multi column printing can be previewed on screen by printing to the screen. When editing pressing SHIFT-CTRL-(will move to the top of the current column and pressing SHIFT-CTRL-) will move to the top of the following column.

If a section of the text is being printed in columnar form (such as this page) it is important to restore the previous ruler at the end of the section, as well as putting in the '>PC 1' command.

18. Proportional Printing

When printing takes place normally, each character occupies a fixed width space on the paper. When Protext right-justifies a line, extra spaces are spread out along the line as evenly as possible. Proportional printing is the process where individual letters occupy different widths (so for example, 'W' takes more space than 'i'). When combined with right-justification, typesetting quality can be achieved. It can only be used with printers which are capable of producing proportionally spaced printing.

Proportional printing can be considered a two part process. One part, controlled by the printer, is the actual printing of the characters. The second part, which determines how many characters can be fitted on a line, is controlled by Protext and uses information about the character widths that is stored in the printer driver. Protext will calculate how many characters the printer is going to be able to fit on a line.

Many dot matrix and daisy wheel printers have only one proportional printing font. The printer drivers for these printers are set up so that the printer control code 'p' will set proportional printing.

Laser and inkjet printers and some dot matrix printers have several proportional fonts available and various control codes or combinations of control codes are used to select these. The easiest way to select fonts is from the Style menu which lists the fonts that are available from the printer driver being used. The 'Protext printer drivers' booklet also gives details of the fonts that are supported on each printer.

How to use proportional printing

(i) The printer must be put into proportional mode. On some printers, particularly daisy wheel printers this may be achieved by means of a switch on the printer. Otherwise an appropriate control should be placed into the document by selecting a font from the Style menu, or by inserting it directly..

(ii) If using a daisy wheel printer a proportional daisy wheel must be used.

(iii) Protext must be instructed to perform the proportional calculations. In most cases printer drivers are defined to use proportional printing automatically.

When this is not the case the document should contain the stored command line '>PP ON' at the start. This applies mainly to 9 pin dot matrix printers, on which microspacing is quite slow. When '>PP ON' is used, sections of the document in fixed pitch are printed with microspacing. If this is not required, '>PP OFF' or '>MS OFF' should be used before the fixed pitch section. The printer driver booklet should be consulted to see whether a particular driver contains the necessary character width table for right justified proportional printing.

Proportional printing and editing

Protext calculates the effect of proportional printing when editing as well as printing. The fonts are not shown proportionally on screen (Protext is not a DTP program), but the correct number of characters are shown on each line and therefore the page breaks on screen are always in the same places as on the printout. When proportional text is used, it is not shown right justified on the screen. Different proportional and fixed pitch fonts may be mixed on any line. A fixed pitch font is treated as a simple example of a proportional font and so different sizes (pica, elite, enlarged, condensed) can be mixed.

Note: Leading spaces in a line can cause problems with alignment of the text. This can be avoided by making full use of the tabs to create lists and to inset the start of paragraphs.

Note: When tabulating data it is recommended that a new ruler line is used with the minimum number of tab stops so that only a single tab character is needed between each column.

Important note:

It is important that the correct printer driver is loaded while editing as well as printing, otherwise the text will be formatted using the wrong character widths and will not correspond to the printout. Tabulated text might also be displayed wrongly if the wrong printer driver was loaded.

Note: To ensure correct centring of text, the '>CE' stored command or a centre tab should be used instead of CTRL-C. The way to centre text in headers and footers is to create a ruler with a centre tab in the middle. On the header or footer line press TAB and then type in the text. The header or footer will always be printed according to the ruler line active when it was defined.

Selecting styles and fonts

Protext keeps track of all changes in fonts and print attributes. Version 5 offers many improvements over earlier versions in this area, including the following:

(i) Different size fonts may be mixed freely in the text (even within a line of text or a footnote) and will be justified or centred correctly.

(ii) Line and page breaks are shown correctly whilst editing using a proportional font or fonts.

(iii) Headers, footers and footnotes are considered separately so that changes to the style of the main text do not affect them.

(iv) The side margin width does not change when a different size font is used.

(v) Centring works in any font. Proportional headers and footers may be centred.

(vi) Tab positions normally depend on the default font and not on the font currently in use. This allows tables to be lined up very easily if different fonts are being used. It is however possible to set tab positions using the current font (by means of the '>FD' stored command) and this is useful in certain cases.

(vii) Everything works equally well with any proportional or fixed pitch font, or any combination of proportional and fixed pitch fonts.

The default font

The default font is a most important concept as it is used to determine many aspects of the way that text is printed, in particular the relationship between ruler lines and the width of the printed text. It is also used to determine the position at which tabulated text is printed and the side margin width.

The default printing font is determined as follows:

(i) If the text contains a stored command '>FD' this will set the default font.

(ii) If the text contains no stored command before the first text line, the default font will be determined by any printer control codes that appear before the first printable character.

The default font is the one that is active when the first printable character is reached. The exception is that if one of the control codes is matched by an off code on the first line of text, only those control codes that appear before this are used (so, for example, a title wrapped in enlarged codes does not change the default font to enlarged).

(iii) If there are none, the default font is taken from the FD command in the printer driver.

(iv) Finally, if this is absent, font 0 in the printer driver is used.

Warning: Note should be taken of (ii), above. If control codes are used incorrectly results that may appear surprising will be obtained. For example if the first line starts with an enlarged code and there is no matching off code on the same line, this will be taken as the default font. The effect will be that the printed line will be twice as wide as normal (it will be 70 enlarged characters wide).

The 'matching off code on the same line' exception means that enlarging a title will not change the default font, but it is important to put the off code on the same line.

In case of difficulties there are two ways to prevent the default font being changed - either use the '>FD' stored command to set the default explicitly or put a space before the first control code.

Examples (bold letters represent printer control codes):

This first line sets the default font as 'rc':

rcThe first line of text

This first also line sets the default font as 'rc', because 'l' is turned off:

rclThe first line of text**l**

This first line sets the default font as 'rcl':

rclThe first line of text

A subsequent '>FD' stored command will change the ruler line and tabs as per the new PW value, but will not affect the side margin width. The side margin will always be calculated using the PW of the initial default font.

Selecting a font

The easiest way to select a font is from the Style menu which is automatically configured to the printer driver being used and shows the available fonts. The menu is divided into two sections. The bottom section contains the standard attributes (bold, italic, etc.) that are present on most printers. The top section (which may extend into submenus) shows the available fonts. These fonts may be considered as 'base' fonts as they may often be modified by one or more of the standard attributes to give different fonts.

The 'Protext printer drivers' booklet accompanying this manual should also be consulted, as this documents the features that each particular printer provides. This booklet may be updated from time to time as new printers are added.

The font menu shows the control codes that are stored in the text for each font. This may be a single code or a combination of codes.

For example, on the Kyocera laser printers the following is a selection of some of the available fonts and the control code combinations used:

h	helve 8 pt
hc	helve 6 pt
hl	helve 14.4 pt
hcl	helve 10 pt bold
r	TMS RM 10 pt
rc	TMS RM 8 pt
rl	helve 14.4 pt

The various fonts provided may be selected from the style menu and a certain combination of control codes will be entered for each font selected. For example selecting Helve 10 point bold will add the three codes h, c and l into the text.

After selecting one of these fonts it may be modified by one of the attributes. For example:

ri	TMS RM 10 pt italic
rcb	TMS RM 8 pt bold

The effect of the attribute depends on the fonts available in the printer, which will generally select the closest available font to the one requested, in a manner which depends on the printer and the printer driver.

The use of combinations of control codes differs from the simple use (when alternate uses of a code turn an attribute on or off). If another font is selected from the menu Protext will change to this font directly - there is no need to individually turn off each control code.

However, control codes may still be individually turned off, as long as the codes are entered in the reverse order to the original codes. For example if 'hcl' is used to select Helve 10 point bold and the next control code use is 'l', this turns off the enlarged code and changes the font to 'hc', i.e. Helve 6 point. This only works with the last used code, not with nested codes. This proviso only refers to control codes used in font selections, the simple attribute codes may be turned on and off at will whether nested correctly or not.

How to print text to a given width

The width of the text is defined by the ruler line width. To obtain the absolute width the ruler width (number of columns) should be multiplied by the average proportional width (PW) for the default font. This PW value is given in inches for each font in the printer drivers booklet.

Some examples:

	PW (inches)
Pica fixed pitch	1/10
Elite fixed pitch	1/12
Star LC24 proportional	30/360
Star LC24 condensed proportional	15/360
Kyocera TMS RM 10 pt	18/300
Kyocera Helve 8 pt	15/300
LaserJet Times Roman 10 pt	21/300
LaserJet Times Roman 10 pt italic	20/300
LaserJet Helvetica 8 pt	16/300

Suppose that a LaserJet is being used with Times Roman 10 point as the default font, and a printed width of 4.2" is required. The PW is 21/300 = 0.07".

The ruler width should be calculated from the formula:

text width = ruler columns * PW

Ruler columns = (text width) / PW

= 4.2 / 0.07 = 60

Thus a ruler line would be defined with the right margin in column 60.

The absolute position of tabs can be determined in the same way.

A note on underlining and tabs

If a section of text is underlined and contains tabs at the start of some lines, these tabs will normally be expanded to spaces and not underlined.

If it is required to underline such a tab, put the underline code immediately before the tab character (on the same line). To underline subsequent leading tabs, turn off underlining at the end of the previous line and put another underline code before the next tab.

It follows from this that to underline a section of text but not the first leading tab the underline code should be placed after the tab character.

Headers and footers

Protext reverts to the default font before printing a header or footer, so that any control codes within the header or footer text are guaranteed to have the same effect, whether or not they are turned off and irrespective of any font changes in the main text.

19. Spelling Checker

What can the Spelling Checker do?

The Spelling checker can do several things. First and foremost it will proof read a document and find most mistakes. This includes simple typing mistakes as well as words that are incorrectly spelt. It will draw attention to these mistakes. It will not, and cannot, automatically correct them. It will check the text very quickly and it provides convenient methods for correcting the mistakes. The spelling checker is a fully integrated part of Protext and can be used without having to leave Protext and load another program.

The spelling checker works in a way that is fundamentally the same as would be used to check spellings using a traditional dictionary. This dictionary, however, is contained on a disc, and whereas normally only words about which there was some uncertainty would be looked up, it may be made to look up every single word in the dictionary, at a speed that could not be approached manually.

Although the spelling checker cannot automatically correct mistakes it does have one feature that goes a long way towards this. In many cases it is able to find the correct word in its dictionary. This is possible because most typing and spelling mistakes can be categorised by a few general rules.

One unavoidable limitation to be aware of is that if a word is misspelt, but the misspelling is also a valid word, then this will not be recognised as an error. A program that did this would be a 'context checker', and development of such programs is included in the research into artificial intelligence being carried out at universities throughout the world.

How the spelling checker decides what is a word

A word is any group of characters starting with a letter, and containing letters and possibly apostrophes or hyphens. Capital letters are distinguished and Protext will check for the correct capitalisation of proper nouns. Dictionaries for various languages are available separately from Arnor, including German and Swedish. Contact Arnor for availability of other languages.

There are additionally a few special cases as follows:

Roman numerals and single letters: Numbers 1-20 (i to xx), and all single letters are contained in the dictionary, as these are often used for numbering paragraphs.

Abbreviations: The full stops in abbreviations are checked, e.g. "e.g.".

Possessives: Words ending with an apostrophe followed by s ("'s") are checked up to the letter before the apostrophe and the possessive ending is ignored. Single apostrophes at the end of a word are also ignored. Words which contain an apostrophe that is not followed by a solitary letter 's' can be stored in the dictionary and will be checked as one word. e.g. "Won't".

Words containing numbers: If the word starts with a number then the whole word is ignored. This means that, for example, dates such as "17th" are not listed as an error. If a word starts with two or more letters and is followed by numbers, then just the numbers are ignored. For example "PC2086" is checked as "PC" (which is in the dictionary).

What sort of errors will it find?

Any words which do not match with a word in the dictionary will be picked up. There are two reasons why the word may not be found.

The word may be misspelt and therefore require correction, or alternatively the word may be a correctly spelt word which is not contained in the dictionary.

The spelling checker can also pick out any words which contain letters in a mixed case, with the exception of words which start with a single upper case letter or contain only upper case letters. For example, 'tHat, 'wHEN' or 'WHAt' would not normally be considered correct and will be selected for possible attention, whereas 'The' would frequently be a quite acceptable use of an upper case letter and will not be singled out. Words that are correctly spelt with a mixture of upper and lower case can be stored in the dictionary in that form.

Another special feature when using the 'SPELL' command is the ability to check for an unknown word separated from a solitary letter by a single space. For example, 'tha t' would be singled out for attention as 'tha' would not be recognised as a correctly spelt word and the letter 't' is separated by a single space. In this case the word and the separated letter are both treated as one unknown word and offered for attention. The reason for this is that this is one of the most frequent forms of typing error, as opposed to spelling error and is simply corrected by removing the offending space.

Dictionary Files

Please note that the supplied dictionary files are subject to copyright laws, and you are expressly permitted to make as many copies as you require, but only for your own use with Protext. There are several dictionary files:

ARNORENG.LEX is the main English dictionary (110,000 words) including the Medical and Legal supplements.
ARNORE1.LEX is a 70,000 word main dictionary without the supplements.
ARNORE2.LEX is a supplementary dictionary
ARNORENG.QIC is the 'quick' dictionary (about 5000 common words)
ARNORE2.QIC is a smaller 'quick' dictionary (about 500 words)
USER.LEX is the user dictionary - created the first time a word is stored
BRTPHON.ENV is a file that defines the procedures for English spell checking. This file must be present in the search path directory.

Note that the correct spell checking language must be configured using CONFIG.

Data is stored in the dictionary file in a compact form. This means that floppy disc users may keep a number of dictionaries on the same disc and still have plenty of room to add new words. Periodically, as words are added to the user dictionary (see later), backup copies should be made for security.

Up to five dictionaries may be specified in the configuration file, plus the quick dictionary which is described below. When a spelling check is carried out each of the specified dictionaries will be checked in turn. One of the advantages of this is that it means that it is very simple to keep special dictionaries for different purposes. For example, it might be desirable to keep foreign language words in a separate dictionary, or to have a special dictionary for technical terms used in a business. Different CFG files could then be created so that the appropriate dictionaries are used. See 'Configuration' for more details.

When a document is being spell checked, it is possible to store words that are correct but not recognised by the dictionary. These words are not inserted into the main dictionary, but are stored in the user dictionary. By default the words will be added to the dictionary 'USER.LEX', though this can be changed. See 'Configuration' for details of how to alter the default update dictionary setting.

A range of commands are available for 'Dictionary maintenance' and these are described fully in the 'Commands' chapter of the manual.

The quick dictionary

A separate file of words called 'ARNORENG.QIC' is also used when spell checking. The words in this file are loaded into memory and checked before the main dictionary, thereby making the spell checking operation much quicker. The quick dictionary is supplied with some of the most commonly used English words. Words may not be added to this dictionary.

Two quick dictionaries are supplied, one that occupies about 2K of memory and contains about 500 words, and one occupying about 20K with about 5000 words. CONFIG may be used to specify which is used - generally the larger would be preferred unless very tight on memory.

Spell Checking a Document

General remarks

Options may normally be abandoned at any time by pressing the ESC key. Pressing the ESC key once pauses the program, at which point pressing ESC a second time will stop it completely and return to command mode. Pressing any other key will continue the program from where it was was paused.

Documents may be checked with up to five dictionaries. Dictionaries are identified by name and the names of all dictionaries to be searched must be specified in the configuration file. By default Protext will use the main dictionaries and the user dictionary ('USER.LEX').

Where to keep the dictionary

When Protext is first installed, a configuration file is created and this will automatically create the necessary default settings for the dictionary files. It will also ensure that, in the case of floppy disc use, the dictionary is copied onto the correct disc.

Two floppy discs:

The dictionary disc should be put in the second drive. The text disc will remain in the first drive. It will then only be necessary to remove the dictionary disc if CONVERT is required.

Single floppy disc:

The text disc will normally be in the drive. To carry out a spell check, remove the text disc and insert the dictionary disc. Press CTRL-S in edit mode or type the command SPELL with no filename. After spell checking replace the text disc and save the corrected version.

Hard discs:

Operation with hard discs is straightforward. The dictionaries should normally be in the 'Protext' directory. CONFIG can be used to specify any extra dictionaries to be searched.

RAM disc:

The dictionary may be copied to a RAM disc if one is present. This is recommended if using floppy discs as it will greatly increase the speed of spell checking. At least 1Mb of memory is required in order to use a RAM disc with Protext. See the appendix on the use of a RAM discs for further details.

Using the Spelling Checker

Text may be checked in three different ways:-

a). All or part of a document during editing.
b). A single word may be checked whilst editing.
c). Complete text files may be checked.

All or part of a document during editing

Pressing CTRL-S will start a spelling check from the current cursor position to the end of the text, or until the check is terminated. Each word in turn is checked against the dictionary, or dictionaries. If the word exists in the dictionary, the following words will be checked until either an unrecognised word is found or the spelling check is abandoned by pressing ESC.

If the word is not recognised for one of the reasons described in above, the cursor will be positioned at the start of the unrecognised word and a dialogue box will display the options available. Words will generally fall into one of two categories:-

A word that is incorrectly spelt, whether because it was mistyped, or is a spelling mistake.

A word that is correctly spelt but is not included in the dictionary.

There are five possible courses of action that may be taken with an unrecognised word, as follows:

C	Change spelling of word
I	Ignore all occurrences of the word
S	Store word in dictionary
L	Look up correct spelling
K	Ignore one occurrence only

If the spelling check is to be terminated at this point, pressing ESC will cancel the spelling check and revert to normal editing.

C - Change spelling of word.

If the word is obviously incorrectly spelt, then C should be selected to correct the word. This has the effect of returning to edit mode so the correction can be made. Once corrected, CTRL-S should be pressed to continue the check. The check will re-commence from the word the cursor is positioned on, or the following word if the cursor is not on a word.

S - Store word in dictionary.

If the word is correct and it is one that should be incorporated into the dictionary, then selecting S will store it in the user dictionary. By default this is 'USER.LEX'. The most obvious examples of words that would be added to the dictionary are technical terms or the words making up the user's name and address.

I - Ignore word.

If the word is correct, but not one that should be stored in the dictionary, then pressing 'I' will cause the word to be ignored and the spelling check to continue. An example of such a word would be 'CTRL' which occurs on many occasions in this manual and is correct in this context, but would not normally be considered a word. Words will be ignored in all subsequent spell checks until a CLEAR or LOAD command is used.

Words that are ignored are held in a buffer in memory. The size of this buffer may be increased using CONFIG.

K - Skip word.

Pressing RETURN causes the word to be ignored on this occasion but another occurrence of the same word will be flagged as a possible error.

L - Look up correct spelling.

L is the most interesting option. If a word is unrecognised and there is some doubt as to whether the spelling is correct, selecting L will get Protext to attempt to find the correct spelling in the dictionary. It will search the dictionary and any likely words will be displayed in a list box for selection. A word can be selected by double clicking on the word, by using the cursor keys and pressing RETURN, or in the case of the first 9 words, directly by pressing the number by the word. If an arrow is shown at the bottom, further words can be viewed by clicking on the line with the arrow or using ↓ or CTRL-↓. If a word is selected, it will be substituted for the mis-spelt word in the text. If ESC is pressed or 'Cancel' clicked on the spelling check options dialogue will return.

The Look up option is by no means guaranteed to find the correct spelling, as the correct word may not be in the dictionary, but in many cases the intended word will be found.

Two methods of lookup are used. Typographical correction is used on the words in the quick dictionary. This is the method previously used and will find words with one letter different, one letter missing, one extra letter or two adjacent letters swapped.

Phonetic lookup operates on both the quick dictionary and the main dictionary. The German dictionary also features compound word correction.

Single word checking

This option permits the checking of individual words during the course of editing a document. Pressing CTRL-Q (for Quick check) will check the spelling of the word on which the cursor is placed. If the cursor is not positioned on a word, the last word prior to the cursor will be checked.

If the word is found in the dictionary, the message 'Word is in dictionary' will be displayed in an alert box and, after a brief pause, normal editing will resume.

If the word is not found, the available options will be displayed in a dialogue box, as described in a) above. In this case, pressing 'S' to store the word, or 'L', to look up the word, function as before, but pressing either 'I' to ignore or 'C' to change will return to edit mode. Once corrected, the word may be re-checked by pressing CTRL-Q again.

Checking an entire text file or a block

This is the option used to check complete documents. It is the fastest of the spell checking options as it reads the file into memory from disc, sorts the words into alphabetical order and then checks them against the dictionary. Whilst the sorting phase takes a few seconds, the speed of checking through the dictionary in alphabetical order results in a quicker check overall.

The command, 'SPELL', followed by the name of the file to be checked, should be entered from Protext command mode. If no filename is given the file in memory will be checked. The command 'SPELLB' will check just the currently defined block. A new status line will appear, giving details of the editing options available.

Checking will start immediately and the first thing that will happen is that messages will appear advising that Protext is 'Reading', 'Scanning' and then 'Sorting'. The words will first be sorted alphabetically. Once sorted, the words can be checked more quickly than would otherwise be possible.

Protext will then start checking the document by comparing the words in the text against the dictionary. Any word that is not in the dictionary will be listed on the screen without stopping at each word.

When the first pass is complete, the second pass will commence and the first unrecognised word will be displayed again, followed by a '?'. There are six options available at this point. 'S', 'I' and 'L' function as described above, except that 'I' causes the word to be ignored for the remainder of the document. The two extra options are 'V' (view context) which simply displays the word in context, and 'B' which moves back to the previously ignored word. Note that when using 'B' all occurrences of an ignored word will be found, unlike when using 'I' to move forward.

Selecting 'C' will result in a few lines of text being displayed on screen with the unrecognised word highlighted in context and the word in question displayed beneath with the cursor on the word, ready for correction. The word may now be edited using the Protext line editing functions as in command mode. See the chapter 'Command mode' for details of these.

If the same word was spelt incorrectly more than once, then when RETURN is pressed to indicate that the correction is complete, the corrected word is displayed in context and the option to change all occurrences is offered. If 'Y' is selected then they will all be changed and the spelling checked again.

If 'N' is selected, the remaining occurrences will be offered for correction, until all of them have been corrected, ignored or the word stored.

If the word only occurred once, then after correction the text will be re-displayed in its new form, with the corrected word highlighted and the spelling of the word will be re-checked.

Note: It is possible to replace the word by two or more words but if this is done, only the first word will be highlighted and only the first word will be checked again.

At the end of the checking operation

Once checking is complete, a number of statistics will be displayed. These are:-

1. Number of words checked.
2. Number of unrecognised words.
3. Number of words changed.
4. Number of words stored.

If the length of any word has been changed, this may have disrupted justification and the text may need to be re-formatted, so a warning message will be displayed as a reminder of this. Finally, the Command mode prompt will return.

If any further editing or formatting is required, the corrected document must first be loaded into Protext. Any reformatting necessary should then be carried out by the normal methods, usually the FORMAT command, before the document is re-saved and printed out.

Large files

The spelling checker is capable of handling any size file, when checking complete files from disc, large files will not be checked in one go. In these cases the file will be checked in two or more parts. When the first part has been checked the next part will automatically be loaded in and checked. Words ignored at any stage will be ignored for the remainder of the document even when checked in more than one section.

Marking words so they will not be checked

It may be desirable to mark a word in the text so that it will be ignored by the spelling checker. This can be done by entering a 'spell ignore marker' immediately before the word. Type ALT-= to obtain the marker.

This may be useful for words particular to one document, or to prevent the 'mixed case' warning appearing.

Spell check whilst typing

Protext is capable of spell checking the text as it is typed. Press SHIFT-CTRL-S to engage this facility. A word will be checked whenever the cursor moves off a word, either by pressing space or a cursor movement key. If the word is not recognised a beep will sound. CTRL-Q can then be used to ignore, lookup, or store the word if required.

Note: Spell check whilst typing is only recommended if the dictionary files are stored on a hard disc or RAM disc.

CONFIG may be used to select spell check whilst typing as the default setting to be used whenever Protext is used.

Spelling Checker Utilities

Protext provides a number of Command mode commands for use with the spelling checker. Most of these commands are concerned with the 'maintenance' of the dictionaries, but others provide the facility to search dictionaries for a word or words.

All these functions may be used by entering the relevant command at the command mode prompt. As with all such commands, any optional parameters may be entered at the same time as the command. If the command is entered on its own, Protext will prompt for the necessary parameters.

Many of the commands allow a dictionary name to be specified. If one is not given, the standard dictionary will be used. By default the commands will not prompt for a dictionary name, but this can be changed using CONFIG.

Menu operation

The commands listed below may all be accessed from the Spell menu.

Utility Commands

The commands, which are described in the 'Commands' chapter are:

ANAGRAM, BUILD, COUNTD, DELWORDS, FINDW, INSWORDS, LISTD, LOOKUP, MAKED

20. Index and Contents

Indexing

All words and phrases that are to appear in the index are marked in the text using 'index word markers' and 'index phrase markers'. Once these markers have been entered they will remain in place in the document enabling a new index to be generated at any time. Text containing index markers may be freely moved around. The index is written to a Protext file with the extension '.idx'.

To index an existing document it will take some time to insert all the necessary markers, but once this is done the work involved in maintaining the index is minimal. It is recommended that index markers are inserted as the text is originally typed as it is clearer then which items are important enough for the index.

Marking a word

An index word marker is entered by pressing:

ALT-W

This can be inserted at the start or end of a word and will appear on screen as an inverse colour asterisk '*'. The word will then appear in the index with a reference to the current page.

Marking a phrase

An index phrase marker is entered by pressing:

ALT-P

One index phrase marker should be inserted at the start of the phrase to be indexed and a second marker at the end. The markers will be shown on screen as inverse video hash symbols '#'. For example:

#Command mode#

will cause the reference to 'Command mode' to appear in the index. Remember to press ALT-P to obtain the '#' markers.

Indexing references that do not appear in the text

Often an index reference is required that is not identical to the words used in the text. For example a section heading might be 'Entering and correcting text' and a reference to 'entering text' is required. The solution is to type a comment line containing the required phrase and mark the phrase on that line. This has the desired effect because the indexer processes comment lines but they are not printed. For example:

```
>>> #Entering text#
```

Index subheadings

Index entries can be made to appear as subheadings to other entries by marking a phrase consisting of the heading, a comma, then the subheading. This would normally be on a comment line as above. For example to create index references as follows:

Delete line 46

and

Line
  deleting 46

the following line would be used:

```
>>> #delete line# #line, deleting#
```

Subsequent entries with the same heading will appear together in the index with the heading only occurring once. For example if a subsequent line in the document was:

```
>>> #line, moving#
```

the index would appear as:

Line
  deleting 46
  moving 47

How to generate the index

The stored command '>GI' should be used at the start of the document. The syntax is:

```
>GI <filename> (<page number string>)
```

For example:

```
>GI manual
```

will cause an index to be generated and written to the file 'manual.idx'.

The optional page number string may contain variable references (like a header or footer) and defines the prefix to the page number, e.g.

```
>GI idxfname &chap&-
```

If the variable 'chap' is defined at the start of each chapter, it will be used in the index references. For example

```
>SV chap = "INTRO"
```

will cause the pages to be numbered "INTRO-1", "INTRO-2", and so on.

The index will be generated whenever one of the print commands are used, i.e. P, PS, PF.

The command INDEX will generate the index and contents but not produce any other output. When using INDEX the '>GI' line may be omitted and the name of the document will be used, with the extension '.idx'. However the index would not then be generated by a PRINT command.

The index file is written as an ordinary Protext document which may be edited and printed with the main document.

Indexing notes

(a) Case sensitivity

Index references are case sensitive. Thus if two references to a phrase are marked they will appear separately in the index unless they are written in the same case.

The exception is the first letter of each index reference which is always converted to upper case.

(b) Sorting

The index is automatically sorted according to the current sorting method. Scandinavian users in particular must ensure that CONFIG is used to select the Swedish sorting method so that Å Ä and Ö are sorted correctly.

(c) Memory usage

Indexing allocates memory dynamically as it operates by building a tree of the index references. If an out of memory message appears when indexing a large document then run Protext again and do not load any files into memory but index the file directly from disc. For example to index a document called 'THESIS' type the following from command mode:

```
INDEX THESIS
```

(d) Hyphenated words

A hyphenated word is treated as a single word for the purpose of indexing.

(e) Format of index

Consecutive page numbers will be grouped, for example a reference which appears on pages 32,33,34 and 35 will be changed to '32-35'.

Entries under a subheading are preceded by a tab. The ruler line written to the index file sets this tab at column 3 but this may be changed if required simply by editing the ruler line in the resulting file.

Contents generating

The table of contents is automatically generated from headings in the text. Headings are identified by being wrapped in particular control codes, which may be specified. Up to three levels of headings can be handled (main headings, subheadings and sub-subheadings).

Unless different control codes are chosen the main headings will be taken as those printed in enlarged (control code 'l') and the subheadings as the bold text. Only two levels are used by default.

To generate a table of contents the stored command '>GC' should be used at the start of the document. This may take various forms depending on the options needed. The simplest form is:

```
>GC <filename>
```

This will generate a two level table of contents using enlarged and bold codes. The table of contents is written to a file with the given name and the extension '.toc'. The command INDEX will generate the index and contents but not produce any other output. When using INDEX the '>GC' line may be omitted and the name of the document will be used. However the table of contents would not then be generated by a PRINT command.

Contents generating options

To specify different control codes, place up to three letters after the filename:

```
>GC <filename> xyz
```

This will generate a three level table of contents using control 'x' to mark the main headings, and 'y' and 'z' the subheadings. A single control must be used to identify each heading level, combinations may not be used.

Further options control the style of the table of contents. By default each contents entry is printed on a separate line with the page number at the right.

The lowest level of subheadings may instead be formatted as a paragraph, with entries separated by commas. To choose this option place a comma after the last control code letter:

```
>GC <filename> x y z,
```

In this style the page numbers can be put in brackets. To choose this put brackets around the control code and comma:

```
>GC <filename> x y (z,)
```

To use brackets but omit the commas:

```
>GC <filename> x y (z)
```

The table of contents is written to a file with a ruler line used to indent the different levels. By default the tabs used for the indenting are 6 columns apart. This can be changed by giving a number in the '>GC' command. The number can appear anywhere after the filename. For example:

```
>GC <filename> x y z 8
```

will use a tab spacing of 8.

Of course the ruler line in the contents file may be altered to change the indent, so this tab value is not binding.

21. Macros and Exec Files

Macros

A macro is a sequence of key presses which can be stored and used at any time with a single key press. The keys used to recall macros are the keys marked 'A to Z' on the main keyboard when used in conjunction with the ALT key and the special function keys on their own and in conjunction with the SHIFT, CTRL and ALT keys.

Protext has 128 expansion tokens, numbered from 256 to 383. 92 of these are allocated to the following keys:-

	f1-f10	257-266
SHIFT	f1-f10	267-276
CTRL	f1-f10	277-286
ALT	f1-f10	287-296
ALT	A - Z	297-322
SHIFT-ALT	A - Z	323-348
	f11,f12	349,353 (if present)
SHIFT	f11,f12	350,354 (if present)
CTRL	f11,f12	351,355 (if present)
ALT	f11,f12	352,356 (if present)

Each of these tokens can be allocated a string of text or codes up to 255 characters long. It should be remembered that the more tokens that are defined, the less memory will be available for text, so it is advisable not to define tokens needlessly.

Macro record mode

This is perhaps the most important section in this chapter. Macro record mode provides an easy method for defining macros, in situations where a repetitive sequence of keystrokes is being used. It is equivalent to using the MACRO command (described below), but much easier to use.

Press CTRL-f1 at any time to turn record mode on. This will work in edit mode or command mode. Then press the key to which the macro is to be assigned. This may be any function key, any function key with ALT, SHIFT or CTRL, or any letter key with ALT or SHIFT-ALT. Pressing ESC at the 'Press macro key' prompt will cancel the command, otherwise record mode will be turned on.

All key presses will now be recorded until CTRL-f1 is pressed again to turn record mode off. This includes all command sequences as well as any letters and symbols. ESC may be used to swap between edit and command mode. The maximum macro length is 255 key presses; if this limit is reached record mode will be turned off.

A special code will be prefixed to each macro definition. This is to identify whether command or edit mode should be selected when the macro starts.

Once a macro has been recorded it can be re-used simply by pressing the chosen function or letter key combination.

It is possible to extend a macro definition by (a) turning on record mode, (b) selecting the same function key, (c) pressing the same function key again and (d) typing the keys to be appended onto the macro sequence.

Notes: a beep will be sounded when CTRL-f1 is pressed as a warning that record mode has been turned on or off. The message 'Macro' will be displayed at the top right of the screen whilst record mode is enabled. If the CTRL-f1 key is redefined (by the MACRO command), the alternative keystroke SHIFT-CTRL-M may be used.

Pre-defined tokens

A number of these tokens are pre-defined to give extra letters such as 'ç' and 'æ' and a few commands such as inserting the date and changing the number of displayed lines. A complete list is given in Appendix 1. These may, however, be redefined by the user if not required for that purpose. The tokens may be listed by the command 'LMACROS' or from the Macro menu.

Care should be taken when defining macros not to use any of the ALT key commands that are needed. If this is done inadvertently the 'Hints and tips' chapter explains how to recover the original functions.

Using macros

Once a macro has been defined it may be used at any time by pressing the appropriate key. Any of them may be used either when in edit mode, or command mode. The most convenient arrangement would probably be to use the function keys for commands which would be used in command mode and the keys 'A' to 'Z' for strings of text to be used in documents.

Macros are called by pressing ALT or SHIFT-ALT and one of the letter keys between 'A' and 'Z', which gives 52 different possibilities. The function keys may be used either on their own, or in conjunction with SHIFT, ALT and CTRL, which gives a further 40 combinations.

When a macro key is pressed, the contents of the string will be entered into the document (if in edit mode), or the command line (if in command mode), as if it had been typed in at the keyboard, and any control codes will be acted upon.

Defining macros using the MACRO command

In most cases record mode will be suitable for defining macros. For some more complex uses the MACRO command may be needed. This takes the form:

```
MACRO <macro key> <macro string>
```

See the 'Commands' chapter for full details.

Saving macros for regular use

Macros may be defined either by record mode or the MACRO command, but once the computer is switched off, these macros will be lost and would require re-entering the next time that Protext was used. In order to preserve the macro definitions so they can be used next time, two commands are provided. These are SAVEKEYS and LOADKEYS.

When one or more macros that have been defined are required to be saved, type the following from command mode:

```
SVK          and press RETURN
```

A message will displayed, something like:

```
SAVEKEYS filename: C:\PROTEXT\PROTEXT.KEY
```

Press RETURN again to save the macros. The filename shown is that from which the macro definitions were loaded. It is not usually necessary to load macros since this will occur automatically when Protext is loaded. Protext will load the macro definitions from a file called 'PROTEXT.KEY' on the search path (usually the PROTEXT directory on a hard disc system). The LOADKEYS command is provided to allow different key files to be used (see 'Commands' chapter).

If the QUIT command is used when macros have been changed but not saved, a warning message will appear 'Save macro changes made (y/n)? '. Pressing Y will cause the changes to be saved.

Exec Files

Exec files are created with Protext to carry out special tasks. An exec file may be used to automate frequently used sequences of commands and to enter sections of text, amongst other things.

What is an exec file?

An exec file is a file that may contain text, commands and codes and which, when called with the EXEC command, will be read by Protext and the contents treated and acted on, as if they had been typed in at the keyboard.

They are created in just the same way as any other text file, but what makes them different is the content of the file and the way it is used later.

Creating an exec file

Creating an exec file is as simple as typing the required text and control codes in, as would be done with any document. Protext will understand the control codes to be instructions to carry out the commands.

In addition to ordinary text, any of Protext's command mode commands may be used. Each of Protext's edit mode commands, such as CTRL-F to format a paragraph and CTRL-<, to jump to the start of the paragraph, have their own code values, as do all the characters which appear on the screen when a key is pressed. These codes are listed in an appendix. It is also possible to insert macro tokens and these are expanded to the contents of the token.

d in a special way, otherwise Protext will consider them
ial 'Escape character' is used to tell Protext that the
/are a code and the escape character used is the
eyboards this is obtained by pressing SHIFT-\. The
ame as used in macro definitions except that the
he circumflex (^). (See MACRO in the 'Commands'

xample exec file to change every occurrence of a certain word to another word in a number of files.

```
L file1
R "PROTEXT" "Protext" GA
S¦13¦
L file2
R "PROTEXT" "Protext" GA
S¦13¦
L file3
R "PROTEXT" "Protext" GA
S¦13¦
```

In the above example, 'L file1' is the command to load a file called 'file1' and 'R "PROTEXT" "Protext" GA' is the REPLACE command used with the options to make it global and automatic. 'S ¦13¦' is the SAVE command. The process is then repeated for 'file2' and so on.

Note: When a new line is used, Protext takes this to mean that a carriage return character (CR) is required, as would normally be given by pressing RETURN after entering the command. In the case of the lines concerned with saving, escape characters have been used to insert an extra CR code into the file. The reason for this is that if a file is to be saved with the same name, then RETURN is pressed once after entering the 'S' and again to confirm the same filename.

Using exec files

Exec files may be used at any time by typing EXEC from command mode, followed by the name of the file to be executed.

The PAUSE command is useful for waiting for a disc to be changed between commands in an exec file (see 'Commands').

Note: It is not possible to 'nest' exec files within other exec files. It is, however, possible to call an exec file from within another but any lines after the EXEC line within the first file will be ignored. An EXEC command would therefore normally only appear as the last line in an exec file.

Using an exec file when loading Protext

Each time Protext is loaded, it is possible for an exec file to be executed and by default the file called 'EXFILE' will be executed. The name of the file to be executed can be changed using CONFIG.

22. Command Mode

This chapter covers a number of topics relating to command mode and commands in general. Full details of all the available commands are given in the 'Commands' chapter.

Command mode Help

Command mode help is available at any time from the Help menu. The menu gives a list of the available subjects, from which the required one may be selected. This will list all the commands relevant to the subject. Alternatively, typing HELP and a command name will give details about the specified command. On a single drive system the file 'COMMAND.HLP' should be copied onto the text disc, if required.

Command entry

Protext has a special feature which permits the entry of commands in a simplified fashion. For example, to save a text file it is only necessary to type 'SAVE' and Protext will prompt with 'SAVE filename:' and wait for entry of a name for the text file.

Alternatively, the parameters of a command may be entered on the same line as the command name, e.g. 'LOAD report', 'SAVE letter'. In this way the commands may be used without the prompts for the parameters appearing, which is often more convenient when familiar with the syntax of the commands. If more than one parameter is specified, they may be separated either by spaces or a commas.

Note: All commands which **require** a parameter will prompt for them if the command is used on its own. Commands which have optional parameters require these to be entered at the same time as the command.

Protext provides a sophisticated line editing facility which is in operation whenever commands are being typed in. If a mistake is made the cursor can be moved back and the mistake corrected in the same way as in edit mode. The line editing commands that are available are listed below.

Line editing facilities

←	Move cursor left one character.
→	Move cursor right one character.
SHIFT-←	Move left one word.
SHIFT-→	Move right one word.
CTRL-←	Move to start of line.
CTRL-→	Move to end of line.
DEL	Delete at cursor.
←DEL	Delete before cursor.
CTRL-A	Alternate characters.
CTRL-←DEL	Delete to beginning of line.
CTRL-E	Delete to end of line.
CTRL-TAB or INS	Switch between insert and overwrite modes.
ESC	Abandon entry of current command.

Protext remembers the previous twenty commands entered at the command mode prompt. These may be recalled with ↑ or ↓ cursor keys. ↓ advances forwards through the buffer, whilst ↑ will recall the last command used and if pressed repeatedly, will recall the earlier commands in turn. When twenty commands have been entered, the first command will be overwritten and so on. When recalling the commands, they will automatically loop again once the beginning or end of the buffer is reached. Once the required command is found, it may be edited in the normal way and the altered command will replace the original in the buffer.

The easiest way to see how this works is to enter a few commands and then use the up and down cursor keys to see the order in which they are recalled.

Abbreviations

Many of the commands can be abbreviated. For example, there is no need to type 'LOAD' in full, typing 'L' will serve the same purpose. Similarly 'S' for 'SAVE' and 'P' for 'PRINT'. A full list of the commands, abbreviations and their parameter syntax, is given in the next chapter and in Appendix 1.

The current filename

After a file has been loaded, or once a piece of text has been saved, the name of the file will be displayed on the status line. This becomes the 'current filename' and is remembered by Protext until changed, either by saving with another name, by use of the NAME command, or by loading a new file. Once a file has a current filename the name may be omitted when saving a file. Entering the SAVE command, and just pressing RETURN when the 'SAVE filename:' prompt appears, will save the file with the current filename. Care must be taken to ensure that it is indeed the correct name, to avoid accidentally erasing something else. If the name displayed is not the correct one, it can be edited.

Directories

ST	*Directories are usually called 'folders' in Atari documentation.*
Amiga	*Directories are usually called 'drawers' in Amiga documentation.*

Protext fully supports the directory tree structure and commands to create and delete directories are provided. It is possible, using CONFIG, to specify directories on which Protext should search for dictionaries, external programs etc. See 'Configuration' for details of how to do this.

Directory paths and drives may also be specified as part of the filename when loading or saving files, in the usual fashion. For example, typing:-

PC/ST:	*LOAD B:\PROTEXT\LETTERS\MYLETTER*
Amiga:	*LOAD DF1:PROTEXT/LETTERS/MYLETTER*
Archimedes:	*LOAD :1.PROTEXT.LETTERS.MYLETTER*

will load the file called 'MYLETTER' which is in the 'LETTERS' directory, which is a subdirectory of 'PROTEXT', which, in turn, is a subdirectory of the root directory of the disc in the second floppy drive.

The current directory is displayed on the banner line. If a file is saved without specifying a full pathname as in the above example it will be saved in this current directory. The current directory can be changed using the command CD. If a file is saved after changing the current directory and a new name is not chosen it will still be saved in its original directory because the full pathname is remembered with the current filename.

Search path and temporary text path

When Protext is installed these two paths will be set up in the configuration file. The STATUS command can be used to display their settings. The temporary text path determines where Protext will save any temporary files it creates while editing large files. This need not be considered further.

The search path specifies a directory in which Protext will search for the following files:

(i) dictionaries, including quick dictionary (if not in current directory), unless a full pathname is given for the dictionary in CONFIG.
(ii) the external programs CONFIG and CONVERT.
(iii) the macro file, PROTEXT.KEY.
(iv) the help file, COMMAND.HLP.

Filenames

PC / ST	*Filenames can be up to 8 characters long, optionally followed by a full stop and a file extension of up to 3 characters.*
PC	*Filenames may not contain spaces or any of the following:* " * + , . / : ; < = > ? [\ ] ¦
ST	*Filenames may not contain spaces or any of the following:* < > ¦ / [ ] , : ; { } * ? " ^ \
Amiga	*Filenames can be up to 30 characters long and may contain any characters except / (the path separator) and : (the device name terminator). It is not recommended that spaces, tabs, commas or semicolons are used in filenames.*
Archimedes	*Filenames can be up to 10 characters long and may not contain spaces or any of the following:* . : * # $ & @ ^ % \

Backup files

Whenever a file is saved the original is preserved as a backup file. The name of the backup file is the same as the original but with the suffix '.BAK' (*except Archimedes - see below*). CONFIG may be used to prevent this automatic creation of backup files, but this is not normally recommended. If a mistake has been made during editing or the new version is accidentally deleted, the backup file may be loaded in order to retrieve the lost text.

See also the section about automatic saving in 'Miscellaneous editing commands'.

PC / ST	*The backup name is created by changing the 3 character file extension. Thus there is only one backup file for two files whose names have the same first 8 characters. For example, suppose there are two files called 'LETTER' and 'LETTER.2'. When 'LETTER' is saved the old version will become 'LETTER.BAK'. If 'LETTER.2' is then edited and saved, the old version of that will become 'LETTER.BAK', replacing the backup of 'LETTER'.*
Amiga	*The Amiga allows filenames up to 30 characters long. The backup name is simply created by appending '.BAK' to the end of the name. If the name is already more than 26 characters long then the backup name will be the first 26 characters with '.BAK' appended. For example, if 'LETTER' is saved the old version will become 'LETTER.BAK'. If 'LETTER.2' is then saved, the old version of that will become 'LETTER.2.BAK' and will not affect the backup of 'LETTER'. If 'LETTER.2.BAK' is then edited and saved the old version of that will become 'LETTER.2.BAK.BAK'.*
Archimedes	*The backup filename is created by prepending 'B!' to the start of the filename. If the original filename is 9 or 10 characters long then the last one or two characters are dropped. For example, if 'LETTER' is saved the old version will become 'B!LETTER'. If 'B!LETTER' is then edited and saved the old version of that will become 'B!B!LETTER'. If 'MANUSCRIPT' is saved the old version will become 'B!MANUSCRI'.*

Changing discs and drives

PC / ST

The currently selected drive is indicated by the letter prefixing the command mode prompt e.g. 'C»'.
Drives may be changed by just typing the drive letter. Drives A to E may be specified by typing the letter without the terminating colon, but any other valid drive can be selected by typing the drive letter followed by the colon. For example, 'F:'. When a drive is changed, the command mode prompt letter will change to suit the new drive. Drives may be catalogued at any time by using the CAT command or pressing f2.

Amiga

The Amiga identifies all discs by name and remembers the name of the current disc. Therefore when a Protext document is saved the Amiga will ask for this remembered 'current disc' to be inserted.

This can initially be confusing when attempting to change a disc and save onto the new disc. The solution is that the new disc has to be 'logged in' before issuing the save instruction. To log in a new disc from command mode use the command 'CD', for example:

CD DF0:
or 0 will log in the disc in drive df0:

When this command is issued the banner line will indicate the new current disc and directory name.

Alternatively using the file selector will log in the disc. Within the file selector it is possible to change drive.

Archimedes

The Archimedes identifies all discs by name and remembers the name of the current disc. Therefore when a Protext document is saved the Archimedes will ask for this remembered 'current disc' to be inserted.

This can initially be confusing when attempting to change a disc and save onto the new disc. The solution is that the new disc has to be 'logged in' before issuing the save instruction. To log in a new disc from command mode use the command 'CD', for example:

CD :0
or 0 will log in the disc in drive 0.

Scroll lock

Commands such as TYPE and PS can output to the screen very quickly and although ESC can be used to pause the display it may be too fast to see what is passing by. For this reason Protext can operate in a 'scroll lock' mode. A screenful is displayed at a time. Press any key (except ESC) to see the next screenful.

PC/Arch.	*The Scroll Lock key is used to turn this mode on and off. This may be used at any time, even while a command is working.*
ST/Amiga	*There is no scroll lock key so the scroll lock function is provided by the keystroke CTRL-HELP.*

Wildcards

PC / ST *Many of the commands allow the use of ambiguous filenames. An ambiguous filename is one which contains 'wildcards'. Protext has two types of wildcards.*

? *may be used to mean 'any single character'.*
* *may be used to indicate 'any number of characters'.*

For example:

DATA?.TXT	*- All filenames beginning with 'DATA' and having one further character (which may be blank), with the suffix 'TXT'.*
B.**	*- All filenames beginning with 'B', of any length and any suffix.*
.	*- All files.*

Note: Only one '' may be used in each part of the filename and suffix.*

Archimedes *Many of the commands allow the use of ambiguous filenames. An ambiguous filename is one which contains 'wildcards'. Protext has two types of wildcards.*

# may be used to mean 'any single character'.
** may be used to indicate 'any number of characters'.*

DATA# - All filenames beginning with 'DATA' and having one further character (which may be blank).
B - All filenames beginning with 'B', of any length.*
** - All files.*

Amiga *Protext uses the standard Amiga wildcards, which are not the same as those described elsewhere in the manual. For full details of Amiga wildcards see the 'AmigaDOS Manual'. For the purposes of this manual, the '?' wildcard works in the same way, but the '*' wildcard should be replaced by '#?'. On the Amiga a '*' is a valid filename character, so if this is used Protext will simply look for a single filename. Amiga filenames may be up to 30 characters long and do not have the same concept of 'file extension'. Instead, any number of full stops may be included in a filename.*

For example:
DATA?.TXT - All filenames beginning with 'DATA' and having one further character (which may be blank), followed by '.TXT'.
B#? - All filenames beginning with 'B', of any length and any suffix.
#? - All files.

Running External Programs from Protext

PC / ST *Protext command mode allows other utilities and applications to be called from it. Most programs, assuming there is sufficient room in memory for both Protext and the other program, can be run in this way. When the program which has been called has been exited, the Protext command prompt will return.*

*In some cases it may be found that a Protext command has the same name as another program or command. If this happens, the Protext command is the one that will be carried out. In order to force the other to be used, the command should be prefixed with an asterisk. For example, to use an external program called 'PRINT', which is a Protext command, type '*PRINT'.*

PC *There are a number of MS-DOS commands which will be found useful when using Protext. They may be used as if they were Protext commands and on completion, the Protext command prompt will re-appear. Some of the more useful commands include: COPY, DISKCOPY and FORMAT. Consult the MS-DOS manual for details. Many of the other MS-DOS commands are provided by Protext anyway such as CD, DEL, MD, RD, REN.*

Amiga *Any other program can be run while using Protext, subject to enough memory being available, by clicking on its icon, or by clicking on the CLI or SHELL window and typing the name of the program. Whilst using AmigaDOS commands or other programs it is often useful to reduce the size of the Protext window so that both windows are visible at the same time.*

Alternatively, if Protext has been run from a CLI window, the 'command mode' of Protext allows other programs to be run from within Protext by typing '' followed by the name of the program. To select command mode simply press ESC from the Protext editing screen. AmigaDOS commands will direct their output to the CLI window and so it may appear that nothing has happened. Click on the back gadget or depth gadget to make the CLI window come to the front and the output will be visible.*

Multi-tasking: programs can be run concurrently with Protext either by clicking on the program's icon, or by using the AmigaDOS command 'RUN'. For example:

```
*RUN CLOCK
```

will install the clock program and return control to Protext. Note that both 'RUN' and 'CLOCK' must be available on one of the discs.

Prefixing a command name with '' will only work if Protext has been run from a CLI window or has been loaded by booting up with the Protext Program disc. If Protext has been run from the Workbench, AmigaDOS commands can be used by (i) clicking on the back gadget or depth gadget so that the Workbench screen becomes visible, (ii) double clicking on the CLI or SHELL icon to open a CLI window, (iii) typing the command into this window. To return to Protext click on the back gadget or depth gadget and click in the Protext window to re-activate it.*

The number and size of programs that can be run from within Protext depends only on the amount of available memory.

Archimedes *Any command or other program can be run from Protext command mode in exactly the same way as from the Command Line. In cases where the command name clashes with a Protext command prefix the command with '*' to ensure the external command or program is run.*

23. Commands

This chapter gives an alphabetical list of the commands available in command mode. A description of the command is given, together with any optional extensions to the basic command, plus details of the command syntax in cases where the command takes parameters that can be entered on the same line.

Note: Letters in brackets, following a command name, are abbreviations or alternatives to the command and may be used instead. Extensions are optional parameters which may be specified to perform variations on the simple command.

The Commands

A *PC / ST only*

Description:	*Selects drive A.*
See also:	*B, C, CHDIR, D, DRIVE, E, 0*

ABANDON (AB)

Description:	This empties Protext's printer buffer, so that it stops sending any more characters to the printer. It also takes the printer out of the 'on STOP' condition. The 'abandon code sequence' is also sent to the printer. (*PC only: If a parallel printer is being used the printer reset line is 'pulsed'*). In many cases the combination of these operations will cause the printer to abandon its own buffer contents, though this is not always possible.
See also:	BACK, CONT, STOP

ACCESS (ACC)

Syntax: ACCESS <ambiguous filename>

Description: Sets the status of a file or files to 'Read-write'. Wildcards are permitted. See PROTECT for details of the reverse operation.

See also: PROTECT

ANAGRAM (AN)

Syntax: ANAGRAM <word-pattern> (dictfile)

Description: Anagram search. This command will search the dictionary for words which match the specified word pattern and display them on screen.

A word pattern can consist of letters, apostrophes and two special 'wildcard' characters. This command can be used to great effect when solving or compiling crosswords.

?	stands for a single unknown letter
*	stands for any number of unknown letters (can be zero).

The following examples show the way in which the word pattern is used.

Word pattern	Words listed
NAME	Anagrams of 'NAME', e.g. 'AMEN', 'MANE', 'NAME'
NAME?	Words containing the letters 'N', 'A', 'M', 'E', and one other, e.g. 'NAMED', 'MEANS'
II????	6 letter words containing two Is
KK*	Words containing 2 Ks (any length)
??????????*	Words of 10 or more letters

See also: FINDW

B *PC / ST only*

Description: *Selects drive B. If drive B is not fitted this causes drive A to be accessed as drive B.*

See also: *A, C, CHDIR, D, DRIVE, E, 1*

BACK

Syntax: BACK ON
BACK OFF

Description: This command turns the background printing feature on or off. The default setting is 'on'.

See also: ABANDON, CONT, STOP

BUILD (BU)

Syntax: BUILD <textfile> <outfile>

Description: Writes unrecognised words from <textfile> to <outfile>. This command carries out a spelling check on the specified file, but instead of offering any unrecognised words for correction, it writes them to the specified file. The words may then be added to a dictionary by using the INSWORDS command.

This is a particularly useful way of adding words which are known to be correct (but are not recognised by the dictionary) to one of the dictionaries. An example of this would be a text file which contained a number of technical words (medical words, technical jargon etc.) which it is desirable to add to a dictionary. This method may be found much quicker than checking the file in the normal way and selecting the Store option for each word.

See also: DELWORDS, INSWORDS, SPELL

C *PC / ST only*

Description: *Selects drive C, if fitted.*

See also: *A, B, CHDIR, D, DRIVE, E*

CALC (CA)

Syntax: CA <expression>

Description: A numeric calculator. Operators +, -, *, / are valid and calculation is by operator priority. Multiplication (*) and division (/) take precedence over addition (+) and subtraction (-), but the order of calculation may be forced by use of brackets round the parts requiring calculation first. For example, '(2+5)*4' will add 2 and 5, then multiply by 4, giving the answer 28, whereas '2+5*4' would multiply 5 by 4 and add 2, giving 22. The result is displayed on screen and may be inserted into the text, at the current cursor position, in one of two ways. Pressing the space bar will insert the integer part of the result. Pressing one of the number keys will insert the result with the number of decimal places specified by the number key pressed. Pressing ESC will not put the result into the document and pressing ESC again will return to edit mode.

CAT

Syntax: CAT (drive:)(path\)(ambiguous filename)

Description: This command displays names and sizes of files that are stored on disc. Optionally, the drive, path and ambiguous filenames may be specified. The filenames are displayed in alphabetically sorted order and the file size is shown, rounded to the nearest K (1024 bytes).

Amiga / Arch. *After changing discs it is necessary to log in the disc first. For example to put a disc into drive 1 and list the files use the command '1' before pressing f2.*

See also: DIR

CHDIR (CD)

Syntax:	CD <pathname>
Description:	Changes the directory to that specified by the pathname.
PC / ST	*Two special options are available. Specifying 'CD \' will select the root directory and 'CD ..' (two full stops) will select the directory from which the current directory branches (the parent directory).*
Amiga	*Specifying 'CD :' will select the root directory and 'CD /' will select the directory from which the current directory branches.*
Archimedes	*Specifying 'CD $' will select the root directory and 'CD ^' will select the directory from which the current directory branches.*
See also:	A, B, C, D, DRIVE, E, MKDIR, RMDIR

CLEAR

Syntax:	CLEAR CLEAR ALL
Description:	Clears the text currently in memory. A request for confirmation is made before this is done. Any words ignored whilst spell checking are 'forgotten' and the default setting of document/program mode is restored. 'CLEAR ALL' clears all files from memory giving a warning if any are unsaved.

CLEARKEYS (CLRK)

Syntax:	CLRK (N)
Description:	All macro definitions are cleared and the default function key definitions are reinstated, unless the 'N' option is given in which case the defaults are not set up. This will not restore any definitions set up by the user - the LOADKEYS command should be used to do this.
See also:	LOADKEYS, MACRO, SAVEKEYS

CLS

Description:	Clears the screen, except for the status line. Useful for mail merge routines and exec files.

CODE

Description:	This waits for a key to be pressed and then displays the token returned by the key. CODE should be used to ascertain the token numbers to be used to include commands in expansion strings when using the MACRO command.
See also:	MACRO

CONFIG (CFG)

Description:	Runs the external program 'CONFIG'. See 'Configuration'.
See also:	LOADCFG, STATUS

CONT (CO)

Description:	This command continues the background printing operation, and starts sending characters from Protext's printer buffer to the printer. It also displays the number of bytes left in the buffer.
See also:	ABANDON, BACK, STOP

CONVERT (CV)

See 'File Conversion'.

COPY *ST / Amiga only*

Syntax:	*COPY <sourcename> (destname)*
Description:	*Copies a file or files to the specified disc and/or path. Wildcards may be used in <sourcename>. If the names are not to be changed, it is not necessary to specify the (destname).*

Examples: *COPY letter A:*
(ST: copies 'letter' from current directory to drive A)
COPY A:\*.* B:
(ST: copies all files from root directory on A to B)
COPY DF1:S/STARTUP-SEQUENCE DF0:
(Amiga: copies a file from the 'S' directory on df1: to df0:)
COPY DF0:#? DH1:
(Amiga: copies all files from root directory on df0: to dh1:)

PC Note: *Use the MS-DOS command 'COPY'.*
Archimedes: *Use '*COPY'.*

COUNT (CT)

Description: This command will count the number of words in the text, excluding stored command lines and their contents.

See also: COUNTB

COUNTB (CTB)

Description: As COUNT, but only counts words in the section of text defined by the block markers.

See also: COUNT

COUNTD (COD)

Syntax: COUNTD (dictfile)

Description: Counts the number of words in the specified dictionary.

Note: The update dictionary is used if no name is specified. The dictionary filename extension '.LEX' may be omitted.

D *PC / ST only*

Description: *Selects drive D, if fitted.*

See also: *A, B, C, CHDIR, DRIVE, E*

DATE *ST only*

Syntax: *DATE <dd-mm-yy>*

Description: *Sets the system date to the specified date.*

See also: *TIME*

PC *Use the MS-DOS command 'DATE'.*

Amiga *Use the AmigaDOS command 'DATE'.*

Archimedes *Use the 'Alarm' program from the Desktop.*

DELWORDS (DW)

Syntax: DELWORDS <textfile> (dictfile)

Description: Delete words in the specified text file from the specified dictionary. This should be used to delete any words which are no longer required, or have been stored in error. If an incorrect word is stored and hence added to the user word file, this file may be edited to remove the word before it is added to the dictionary file. DELWORDS would not be needed in that case.

See also: BUILD, INSWORDS

DIR

Syntax:	DIR (dr:)(path\)(ambig filename)
Description:	Performs a directory listing of the specified files, in the (optionally) specified path of the (optionally) specified drive.
See also:	CAT

DOC

Description:	Selects document mode. This is the normal mode used when creating documents with Protext. By default, when Protext is loaded initially, it is the mode selected. When document mode is in use, all stored commands, ruler lines and printer control codes are obeyed at the time of printing or formatting.
Note:	It is possible to specify filename extensions which will always cause any file with that extension to be loaded as a document. This is done by means of the CONFIG program.
See also:	PROG

DRIVE (DR) *PC / ST only*

Syntax:	*DRIVE <dr>*
Description:	*Selects specified drive letter. Drives may also be selected by typing the letter followed by a colon. For example, 'DRIVE F' is the same as 'F:'.*
Amiga / Arch.	*Use the command CHDIR.*
See also:	A, B, C, CHDIR, D, E

E *PC / ST only*

Description: *Selects drive E, if fitted.*

See also: *A, B, C, CHDIR, D, DRIVE*

ECHO

Syntax:

```
ECHO ON
ECHO OFF
```

Description: 'ECHO OFF' suppresses the echoing of command lines onto the screen. 'ECHO ON' restores the display. These commands are useful within EXEC files.

See also: EXEC

ERASE (DEL)

Syntax:

```
DEL (dr:)(path\)(filename)
```

Description: Deletes the specified file or files from the directory. Optionally the drive and/or path may be specified. Wildcards may be used, but care is required as confirmation is only required when '*.*' is used to delete all the files in the specified directory.

See also: RENAME

EXEC (X) Execute a file of commands

Syntax:

```
EXEC <filename>
```

Description: The file specified will be opened for reading and the contents will be treated as if they were input from the keyboard, until the end of the file is reached, at which time normal operation will resume.

See also: CODE, ECHO, EXECS, PAUSE

EXECS (XS) Execute a string of commands

Syntax: EXECS <string>

Description: The same as EXEC except that the input is taken from the string supplied as parameter to the command, instead of from a file.

See also: EXEC

EXT

Syntax: EXT <str>

Description: Sets the default filename extension for loading and merging files. <str> may be any combination of up to three alpha-numeric characters. For example: 'DOC', 'TXT' or '123'.
If no filename extension is given when a file is specified for loading, Protext will initially search for a file of the name typed, but if not found, it will search for a file with the extension set by this command.

See also: LOAD

FF Form feed.

Description: A form feed character is sent to the printer. This is particularly useful with laser printers, or to eject the paper from a printer when form feeds are not enabled.

FIND (F)

Syntax: FIND <string> <options>

Description: Searches for a string of text. See 'Find and Replace'.

See also: FINDB, REPLACE

FINDB (FB)

Syntax: FINDB <string> <options>

Description: Searches for a string of text within the current block. See 'Find and Replace'.

See also: FIND, REPLACEB

FINDW (FW)

Syntax: FINDW <word-pattern> (dictfile)

Description: Searches the dictionary for a match to the word pattern. It is used in a similar fashion to ANAGRAM, the difference being that FINDW will only select words with the characters in the same order and position in the word that was specified in the word-pattern.

A word pattern can consist of letters, apostrophes and two special 'wildcard' characters. This can be used to great effect when solving or compiling crosswords.

?	stands for a single unknown letter
*	stands for any number of unknown letters (can be zero).

This is most clearly explained with examples:

Word pattern	Words listed
BA???	5 letter words starting with 'BA'
A?A?U?	6 letter words with 'A' in the first and third positions and 'U' in the fifth position
BA*	All words starting with 'BA' (any length)
S*K	Words starting with 'S' and ending with 'K'
*ALE	Words ending with 'ALE'
????????????	12 letter words
*a*e*i*o*u*	Words containing all 5 vowels in alphabetical order
*	All words in the dictionary!

See also: ANAGRAM

FIX

Description: When this command is used, all tabs and soft spaces are converted into hard spaces. Soft returns are converted into hard returns.

See also: FIXB

FIXB

Description: As FIX, but only affects the currently defined block. This command is particularly useful for 'fixing' sections of text that it is important will not be disturbed when formatting takes place. If the section of text is 'fixed' first, all tabs are removed. A second use for this command is in conjunction with text which is to be typeset and which it is important retains the precise layout.

See also: FIX

FORMAT (FT)

Description: This command will reformat the entire document according to the margins, rulers, tabs and stored commands within the document.

PC Note: To format a disc using the MS-DOS FORMAT command, the drive letter should be specified, e.g. 'FORMAT A:'.

See also: FORMATB

FORMATB (FTB)

Description: As FORMAT, but only affects the currently defined block.

See also: FORMAT

FSORT

See 'File Sorting'.

GOTO (G)

Syntax: GOTO (option) <number>

Description: Moves the editing cursor to the specified line,column or page in the text. The number should be prefixed by the letter 'P' for page, 'L' for line or 'C' for column. If the letter is omitted, the default is line.

Note: If line is selected, the line is the 'absolute' line in the text, not the line number within the page, which ignores stored command lines.

Note: If the page option is selected and Protext is not in page mode, the cursor will not be moved.

HELP (H) or f1

Syntax:
HELP
HELP <subheading>
HELP <command name>

Description: The Help menu displays the subheadings that are available.

The HELP command may be entered with the appropriate subheading to list all commands relating to that subject, together with any abbreviations. Typing HELP and a command name will give details of the syntax of the command specified. The file COMMAND.HLP must be present in the search path directory in order to use HELP.

Example: HELP FILES

INDEX (IX)

Syntax: INDEX (<infile>)

Description: Generates the index and table of contents for the current text or a file, without printing it. See 'Index and contents'.

INK *ST only*

Syntax: *INK <ink number> , <colour>*

Description: *Assigns a colour to one of 4 ink numbers. Ink numbers are between 0 and 3. The colour takes the form '&rgb', where r, g, and b are numbers representing the amount of red, green, and blue (between 0 and 7). Setting colours is a two stage process. INK is used to choose 4 colours, then CONFIG is used to assign these inks to particular uses (see UTIL 4).*

After using INK, Protext restores the Desktop ink settings on quitting.

In high resolution mode there are only two possible colour settings, white on black and black on white. These are selected by the commands 'INK 0,1' and 'INK 0,0' respectively. There is also an option in CONFIG to set the mono screen colours. This should be used in preference to the INK command which would also change the Desktop menu colours when used within Protext.

Note: *The border colour is always INK 0 in colour modes and always black in high resolution monochrome mode.*

Example: *INK 1,&070* *sets ink 1 to be bright green*

PC *Use CONFIG to set colours.*

Amiga *Amiga - use Preferences to set 'Workbench colours'. CONFIG will then use these colours.*

Archimedes *Use CONFIG to set the colours for the screen mode chosen to run Protext in (4, 16 or 256 colours). If it is required to run Protext in modes with different numbers of colours available a separate configuration file should be set up for each type of mode.*

INSWORDS (IW)

Syntax: INSWORDS <textfile> (<dictfile>)

Description: All the words in the specified text file are incorporated into the specified dictionary. This command would normally be used to add all the words in a text file to the specified dictionary.

The text file would often be one created with the BUILD option, or a normal text file containing the words to be added.

See also: BUILD, DELWORDS

KEY See MACRO.

KEYB

Syntax: KEYB <keyboard type>

Description: Selects a keyboard type.

PC *Types are: 1. XT, 2. AT (the same as 1), 3. extended AT, 4. alternative AT layout. CONFIG would normally be used to select keyboard type.*

ST *Types are: 1. normal, 2. numeric keypad functions as on PC.*

Amiga *Types are: 1. standard Protext, 2. numeric keypad functions as on PC, 3. standard Intuition.*

See also: LANG

KEYDEF

Syntax: KEYDEF <key number> <list of tokens>

Description: Redefines a key. The tokens set the following: the key unshifted, with SHIFT, with CTRL, with ALT, with NUM LOCK on, with CAPS LOCK on, with SHIFT and ALT, with SHIFT and CTRL. If less than 8 tokens are given only those shift states given are changed. Keys would normally be defined using CONFIG as this is easier.

LANG

Syntax: LANG <keyboard language>

Description: Selects the keyboard layout for a given country. Languages are: 1. US, 2. UK, 3. France, 4. Germany, 5. Spain, 6. Italy, 7. Sweden, 8. Denmark, 9. Norway, 10. Portugal. Languages can also be selected by pressing SHIFT-ALT and the function key of the language number. For example SHIFT-ALT-f4 is the same as 'LANG 4' and can be used to temporarily use the German keyboard layout. CONFIG would normally be used to select keyboard language.

See also: KEYB

LINK

Syntax:

```
LINK ON
LINK OFF
```

Description: When the screen is split into two windows, 'LINK ON' links the two files so that scrolling up and down with SHIFT-↑ and SHIFT-↓ scrolls both windows simultaneously. This function is also available from the Swap menu.

LISTD (LD)

Syntax: LISTD (dictfile) (word) (outfile)

Description: This command causes Protext to list all the words in the dictionary, starting from <word> or the start if this is omitted.

If no dictionary name is given, the contents of the update dictionary (specified in the configuration file) will be listed. Listing is to the screen unless (outfile) is specified, in which case the listing will be sent to a file of that name. It is not possible to list the main dictionary (but see FINDW and ANAGRAM).

LMACROS (LM)

Description:	Use of this command will list the contents of all the defined macros, with both key descriptions and token numbers. Control codes are displayed in escape code form, e.g. ^9^.
See also:	MACRO

LOAD (L)

Syntax:	LOAD <filename>
Description:	A document will be loaded into memory from a disc file of the specified name. If a file does not exist with the specified name, Protext will search for a file of the same name, but with the filename extension set by the EXT command. A warning message will be given if the text currently in memory has not been saved. Press 'Y' to confirm that this text is to be discarded. Once loaded, the specified filename will become the 'current filename'. Depending on the settings in the configuration file, the file may be loaded as a program file, or as a Protext document file.
Note:	If the command name is entered on its own, Protext will prompt for a filename. If RETURN is pressed at this point the file selector will be called up to allow selection of a file.
See also:	EXT, MERGE, SAVE

LOADCFG (LCFG)

Syntax:	LOADCFG <filename>
Description:	Different configuration files may be loaded at any time. It may be found useful to have a number of configuration files for different purposes. For example:- A CFG file specifies the default printer driver, default dictionaries to check, page dimensions, amongst many other things. It is therefore possible to have different CFG files for manuscripts (which might require double spaced text on continuous stationery in draft mode) and letters (which would require single sheet and a daisywheel printer). If different CFG files were saved as MANSCRPT.CFG and LETTER.CFG, then using this command to load the appropriate CFG file would automatically configure Protext for the intended purpose. Most options set in the CFG file are acted upon. Exceptions are the temporary path and the sizes of memory allocated for text, which are only changed when Protext is re-loaded.
See also:	CONFIG, STATUS

LOADKEYS (LDK)

Syntax:	LDK <filename>
Description:	The macro definitions are loaded from a file, with the specified name and the extension '.KEY'. Any existing macros on the same keys are overwritten, but those on keys not defined in the file remain. To clear all the old macros use CLEARKEYS first. If no filename is given (by just pressing RETURN in response to the prompt), the key file that was last loaded will be used. This is useful for restoring the macros after changing them temporarily or making a mistake whilst recording a macro. Normally this will be the file 'PROTEXT.KEY' which is loaded from the search path when Protext is first run.
See also:	CLEARKEYS, MACRO, SAVEKEYS

LOOKUP (LK)

Syntax: LOOKUP <word>

Description: This command searches all the dictionaries for similarly spelt words to the word specified. It is equivalent to the 'L' option which is available during normal spell checking.

MACRO (KEY)

Syntax:
```
MACRO (char)<letter/number> <string>
KEY   (char)<letter/number> <string>
```

Note: There should be no space between 'char' and 'letter/number' if 'char' is included, but there should be a space between 'letter/number' and 'string'.

Description: MACRO and KEY are alternative names for the same command. This command allows temporary strings to be created at any time. The command is used from command mode.

'char' should be one of 'S', 'C', or 'A' (for SHIFT, CTRL or ALT) for a function key or 'S' if the key to be defined is an alpha key between A and Z.

'letter/number' must be a letter between A and Z (to select an alpha key), or a number between 1 and 10 (to select a function key).
'string' is the string of text, or codes, and should be wrapped in quotation marks.

Examples:

```
MACRO A "Hello"                - Pressing ALT-A will give 'Hello'.
KEY SA "Goodbye"               - SHIFT-ALT-A will give "Goodbye".
KEY 9 "CD \PROTEXT^13^"        - Pressing f9 on its own will change
                                 to the Protext directory.
KEY C9 "COPY A:"               - CTRL-f9 will enter 'COPY A:' and
                                 wait for further text to be entered.
```

Note: Macros may be up to 255 characters in length, including the command name and the key letters/numbers. It is, however, possible to link one expansion token onto the END of another so that a macro may be longer than 255 characters. for example:-

```
MACRO 383 " ... this is the continuation string"
MACRO A "This is the start of the string ... ^383^"
```

will result in macro A continuing with macro 383.

Note: A macro string may contain any characters and control codes. Any normal text may be typed in from the keyboard as usual, but in order to be able to enter control codes, an escape code must be used to inform Protext that the characters which follow constitute a control code. The escape code used by Protext is the caret or circumflex (^). This is obtained by pressing the number 6 key on the main keyboard, whilst holding down the SHIFT key. It is used to allow entry of the following:

^<number>^	Inserts the code specified by the number. The code may be entered in decimal, e.g. ^13^, or hexadecimal, in which case it must be prefixed by either # or &, e.g. ^&0D^. The CODE command should be used to determine the codes that need to be used.
^<letter>	is translated as a control code between 1 and 26 e.g. ^A would be translated as Ascii code 1, ^B as 2, etc.
^^	is translated by Protext as a single circumflex. This must be used if a '^' is required in the string.

Note: If it is required to cancel a key definition, this can be done by using a null string ("") following the key letter in the parameters. This may prove useful to discard macro definitions which are no longer required, saving memory.

See also: CLEARKEYS, CODE, LOADKEYS, SAVEKEYS

MAKED (MKD)

Syntax: MAKED <textfile> <dictfile>

Description: Creates a new dictionary file containing all the words from <textfile>. If no filename extension is specified for <dictfile>, '.LEX' will be added to the name. When a new or additional dictionary is required, this option should be used to create it.

Note: This command is similar to INSWORDS, the difference being that INSWORDS will only add words to an existing dictionary, whereas MAKED will create a new dictionary with the specified name.

MERGE (M)

Syntax: MERGE <filename>

Description: This is similar to LOAD but whereas LOAD clears any existing text from memory and then loads the file in, MERGE inserts the new file into the existing text at the current cursor position. Protext will search for a file of the specified name and if not found, it will search for a file of the same name, but with the filename extension specified by EXT, or set within Protext.

Note: Care should be taken to ensure that the cursor is in the required position before using this command.

Note: The current filename is NOT changed.

See also: LOAD

MKDIR (MD)

Syntax: MD (dr:)(path\)(dirname)

Description: Creates a new subdirectory, optionally on the specified drive, optionally from the specified drive path and with the name specified by 'dirname'.

See also: CHDIR, RMDIR

NAME (N)

Syntax:	NAME <filename>
Description:	Changes the name of the document in memory. The new name becomes the current filename.

NUMBER (NUM)

Description:	The purpose of this command is to add line numbers to, or remove line numbers from, the beginning of every line of text. This command will prompt for whether numbers are to be added or removed from the document. If adding line numbers is selected, a starting line number and the value by which each subsequent number is to be incremented will be requested. Amongst other uses, this provides a convenient method of writing BASIC programs in Protext.
See also:	NUMBERB

NUMBERB (NUMB)

Description:	This command is similar to NUMBER, but only adds or removes numbers within the marked block.
See also:	NUMBER

PARALLEL (PAR) *PC / ST only*

Syntax:	*PAR (num)*
Description:	*Selects the parallel (Centronics) printer port for the output of all printing, and loads the printer driver for the specified port. The printer driver for a given port can be configured using CONFIG. If no printer driver has been configured then the simple printer driver will be selected. 'num' is the number of the parallel port to be used. If no number is specified then 1 is used. This is the parallel port that is built in to most computers. Ports 2 and 3 are the optional additional printer ports which may be fitted* (PC only).

See also: *PRINTER, SERIAL*

Amiga Note: *Use Preferences to select the printer port.*

Archimedes: *Use the Configure application from the Desktop (or *CONFIGURE) to select the printer port.*

PATH *ST only*

Syntax: *PATH <string>*
Example: *PATH c:\protext;c:\utils;c:\*

Description: *The PATH command is equivalent to 'SET PATH=<string>' and is used for setting a search path used when Protext is asked to run an external program. The string is a list of directory paths separated by semicolons.*

PAUSE

Syntax: PAUSE

Description: Causes Protext to go into a 'waiting' condition. When this command is read by an EXEC file, the program will halt until a key is pressed. Optionally a message will be displayed. This is useful during the course of an EXEC file being executed, as it will permit discs to be changed and messages to be displayed before continuing execution.

See also: EXEC

PM

Syntax: PM
Description: Print menu. This command calls the print dialogue box from which the various printing options may be selected.

PRINT (P)

Syntax: PRINT (num)

Description: This command permits printing of the document in memory. By default, the printing will be draft quality unless the appropriate NLQ printer control codes have been used in the text. All stored commands will be obeyed if document mode is being used, but will be printed if program mode has been selected. Optionally the number of copies to be printed may be specified.

Extensions: A filename.

Syntax: P <infile> (num)

Description: Optionally a filename may be specified after the PRINT command, in which case printing will take place directly from the file of that name on disc and a number may be specified for the number of copies to be printed. It will not be loaded into Protext and will not affect any text in memory.

See also: PRINTB, PRINTF, PRINTP, PRINTQ, PRINTS

PRINTB (PB)

Description: Only the section of text defined by the block markers will be printed, but stored commands in the text are taken into account.

See also: PRINT

PRINTER (PR)

Syntax: PRINTER <filename>

Description: Loads a printer driver (see below) to configure Protext for use with another printer. The '.PPD' extension in the filename need not be typed. A number of printer drivers are supplied and extra printer drivers may be created to suit printers not covered by the supplied drivers by modifying the supplied drivers.

See also: PARALLEL, SERIAL

PRINTF (PF)

Syntax: PRINTF <outfile> <infile>

Description: Sends the contents of the document in memory, including all printer control codes, to a disc file instead of the printer.

PC *This file can later be printed using the MS-DOS 'COPY' command.*

For example:

COPY printfile PRN

Note: All printer control codes are sent to the file in the same form as if they were being sent straight to the printer. This means that when the file is printed, all codes will be acted on by the printer, including those to pause the printer at the end of each page (if supported by the printer) so that new paper may be inserted.

See also: PRINT, PRINTFB, SAVE, SAVEA

PRINTFB (PFB)

Syntax: PFB <outfile>

Description: Only the section of text defined by the block markers will be sent to the file, but all stored commands in the text are taken into account and all control codes are sent to the file.

See also: PRINTF, SAVEAB

PRINTOFF (PROFF)

Description: Cancels the echoing of screen output to the printer, which has previously been initiated by use of the PRINTON command.

See also: PRINTON

PRINTON (PRON)

Description: All output to the screen will be echoed to the printer, after this command has been used, until PRINTOFF is used to cancel it.

See also: PRINTOFF

PRINTP (PP)

Syntax: PP (<list of numbers>)

Description: Print selected pages. If no numbers are given a prompt appears for each page allowing it to be printed or skipped. If a list of numbers is given, just those pages are printed. The list may include individual numbers and ranges of numbers, such as '13-16', separated by commas.

Extensions: A filename.

Syntax: PRINTP <infile> (<list of numbers>)

Description: Prints text from a file on disc, but permits pages to be skipped. Optionally a list of pages to be printed may be specified.

See also: PRINT, PRINTPQ, PRINTPS

PRINTPQ (PPQ)

Syntax: PPQ (<list of numbers>)

Description: As PRINTP, but printing takes place in Near Letter Quality mode, permitting pages to be skipped. Optionally a list of pages may be specified. See note under 'PRINTQ'.

Extensions: A filename

Syntax: PPQ <filename> (<list of numbers>)

Description: As above, but prints text in Near Letter Quality mode from a file on disc. Optionally a list of pages to be printed may be specified.

See also: PRINTP, PRINTQ

PRINTPS (PPS)

Syntax: PPS (<list of numbers>)

Description: This option is similar to the PRINTS option described above, the only difference being that the option to view each page, or print a list of pages is available, as in the PRINTP command.

Extensions: An input filename.

Syntax: PPS <infile> (<list of numbers>)

Description: As PS, but with the option to skip or print a list of pages.

See also: PRINTP, PRINTS

PRINTQ (PQ)

Syntax: PQ (num)

Description: The effect of PRINTQ is to cause printing to take place in 'Near letter quality' (NLQ), without the need to insert the NLQ control code into the text. Otherwise it is exactly the same as PRINT. Optionally the number of copies to be printed may be specified.

Note: This command will only work correctly if a suitable printer driver is being used and the printer is capable of NLQ output.

Extensions: An input filename.

Syntax: PQ <infile> (num)

Description: Printing will take place directly from the file of that name on disc. It will not be loaded into Protext and will not affect any text in memory. Optionally the number of copies to be printed may be specified.

See also: PRINT, PRINTPQ, PRINTQB

PRINTQB (PQB)

Description: Serves the same purpose as 'PRINTQ', but only on a block of text selected by use of the block markers.

See also: PRINTQ

PRINTS (PS)

Description: The file in memory will be printed to the screen in the same format that it would be printed on the printer. Stored commands are obeyed. This option provides an easy way to see how page breaks, headers and footers etc. will appear.

Note: It is also particularly useful when used in conjunction with a mail merging file, to ensure that the expected results are being obtained, before printing.

Extensions: An input filename.

Syntax: PS <infile>

Description: This command allows printing to the screen directly from a file. The file will be printed directly from disc. It will not be loaded into Protext and will not corrupt any text in memory. It provides a convenient method of reviewing the contents of a disc file without the need to load it into Protext.

Note: Pressing ESC once will pause printing. Pressing ESC a second time will abort further printing, but pressing any other key will continue printing. If printing is aborted by pressing ESC twice during a PRINTS of a file in memory, on returning to edit mode the cursor will be positioned at the point that had been reached in the text. This makes it easy to correct mistakes noticed whilst printing to the screen.

See also: PRINT, PRINTPS, PRINTSB, TYPE

PRINTSB (PSB)

Description:	Only the section of text defined by the block markers will be printed to the screen, but all stored commands in the text are taken into account.
See also:	PRINTS

PROG

Description:	Selects program mode. The word 'Program' will replace 'Document' on the status line. Program mode is particularly useful for writing source code for assemblers and compilers, and for editing data files. Tabs are set at every eighth column (but can be changed using the TAB command or with CONFIG) and there is no right margin or formatting. When program mode is in use, most of the normal editing commands may be used in the same way that they are used in document mode, but it should be noted that because there is no formatting, any of the commands concerned with formatting of the text, rulers and pages, will have no effect. Appendix 1 gives details of which commands are not available.
Note:	When program mode is in operation, any stored commands and ruler lines are printed and NOT obeyed. This provides a means of printing out a text file complete with stored commands, rather than them being acted on.
Note:	It is possible to specify filename extensions which will always cause any file with that extension to be loaded in program mode. This is done by means of the CONFIG program.
See also:	DOC, TAB

PROTECT (PROT)

Syntax: PROTECT <ambiguous filename>

Description: Sets the status of a file or files to 'Read-only'. Wildcards are permitted. Files which have read only status can not be overwritten by saving or copying a file of the same name. An error message will be given if an attempt is made to do so. Protected files are indicated in the catalogue by an asterisk following the filename. See ACCESS for the reverse operation.

Note: PROTECT cannot stop files being erased if the disc is reformatted, or a complete disc is copied onto the disc.

See also: ACCESS

QUIT (Q)

Description: Quits Protext. If a document is in memory and any changes have been made to it since it was loaded or last saved, a caution will be issued, warning that the document has not been saved and asking for confirmation to proceed.

RENAME (REN)

Syntax: REN (dr:)(path\)<oldname> <newname>

Description: Renames the file called oldname to the new name. Only the file name is changed. Neither the contents nor their location on the disc are affected.

See also: ERASE

REPLACE (R)

Syntax: REPLACE <find-string> <replace-string> <options>

Description: Searches for a string of text and replaces it with the second string. See 'Find and Replace'.

See also: FIND, REPLACEB

REPLACEB (RB)

Syntax: REPLACEB <find-string> <replace-string> <options>

Description: Searches for a string of text within the current block and replaces it with the second string. See 'Find and Replace'.

See also: FINDB, REPLACE

RMDIR (RD)

Syntax: RD (dr:)(path)(dirname)

Description: Removes the named directory from the drive.

Note: The directory cannot be removed if it contains any files and it must not be the selected directory.

See also: CHDIR, ERASE, MKDIR

SAVE (S)

Syntax: SAVE <filename>

Description: The complete document in memory will be saved to a disc file with the name specified.

Note: If only the command name is entered, Protext will prompt for a filename. If the file already has a current filename, then pressing RETURN will result in the file being saved with the same name. Alternatively, a new name may be specified, which will then become the current filename. If a new name is chosen which conflicts with an existing file a warning message will be given requesting confirmation before overwriting the file.

Amiga / Arch. *To save a document on a different disc, either log in the disc as described earlier before saving the file, or specify the drive or disc name as part of the file name, for example:*

SAVE DF0:LETTER (Amiga)
SAVE :0.LETTER (Archimedes)

See also: LOAD, PRINTF, SAVEA, SAVEB

SAVEA (SA)

Syntax: SAVEA <outfile> <infile>

Description: Saves the text in ASCII form. The text in memory will be sent to a file on disc with the specified name, instead of to the printer. Tabs are expanded to spaces. Each line is terminated with a carriage return. By default there are no margins, as if the stored command '>ZM' was be inserted at the start of the file. The file will be a 'pure' ASCII file, suitable for use by other programs, or for transmission by 'Electronic Mail'.

The CONVERT utility provides the option for reformattable ASCII files. The difference is that these do not contain carriage returns at the end of lines (only at the end of paragraphs) and are thus suitable for loading into other word processing or DTP software.

See also: CONVERT, PRINTF, SAVE, SAVEAB

SAVEAB (SAB)

Syntax: SAB <outfile>

Description: Save block as ASCII file. Only the section of text defined by the block markers will be sent to the file, but all stored commands in the text are taken into account.

See also: PRINTFB, SAVEA

SAVEB (SB)

Syntax: SB <filename>

Description: This is the same as SAVE except that only the text within the block defined by the block markers is saved.

Note: The current filename is NOT changed.

See also: SAVE, SPLIT

SAVEKEYS (SVK)

Syntax: SVK < filename >

Description: The currently defined macro definitions are saved in a file, which is given the extension '.KEY'. If no filename is given, the name of the previously loaded key file will be used.

Note: This file is stored in ASCII form as a series of KEY commands and so may, with care, be edited. The numbers used in the KEY commands are the expansion tokens and are the numbers returned by the CODE command when a function key or ALT + letter combination is pressed.

See also: CLEARKEYS, MACRO, LOADKEYS

SERIAL (SER) *PC / ST only*

Syntax: SER (num)

Description: Redirects all printed output to the serial interface, for use with a serial interfaced printer, and selects the configured or the simple printer driver (see 'PARALLEL', above). 'num' is the number of the serial port to be used, and may be 1 for the built in RS232 interface or 2 for the second port (if fitted). If no number is specified, the default port is 1.

Note: It may be necessary to configure the computer for the serial printer by setting the baud rate and other options.

PC *The MODE command should be used to configure the serial port.*

ST *The Gem desktop accessory CONTROL.ACC should be used to configure the serial port.*

See also: PARALLEL, PRINTER

Amiga *Use Preferences to select the printer port.*

Archimedes: *Use the Configure application from the Desktop (or *CONFIGURE) to select the printer port.*

SET *ST only*

Syntax:	*SET <variable> = <string>* *SET ; lists all set variables*
Description:	*The SET command is provided to allow the setting of environment variables. This will be useful to those using Protext as a shell for running other programs. Note that environment variable names are case sensitive.*

SHOW (SH)

Syntax:	SHOW
Description:	Lists all the currently open files showing the current name of each, the size and the original full pathname. Any file that has been changed and not saved is marked with a tick.

SPELL (SC)

Syntax:	SPELL <filename>
Description:	Spell checks the named file. If no filename is given the current text is checked. See 'Spell Checking'.
See also:	SPELLB

SPELLB (SCB)

Description:	Spell checks the currently defined block. See 'Spell Checking'.
See also:	SPELL

SPLIT

Syntax:	SPLIT <filename> <size>
Description:	Excessively large files may be broken down into a number of smaller files using this command. 'filename' is the name of the file to split and 'size' is the required size of the new files, specified in bytes. The new files will be given the same name as the original file, but with an incrementing filename extension. For example:- SPLIT BIGFILE 20000 will result in a number of files called 'BIGFILE.1', 'BIGFILE.2' and so on. Each file will be approximately 20,000 bytes long, except for the last file.
Note:	Protext will not split the file at exactly the size specified, as it will always attempt to split the file at the start of a new line.
See also:	SAVEB

SPOOL (SPON)

Syntax:	SPOOL <filename>
Description:	All output to the screen will also be sent to a file on disc with the specified name until the file is closed with the SPOOLOFF command.
See also:	SPOOLOFF

SPOOLOFF (SPOFF)

Description:	Cancels the echoing of all screen output to a file, having first closed the file.
See also:	SPOOL

STATUS (STAT)

Syntax: STATUS

Description: This command lists the most useful of the configuration file settings, such as the dictionaries to be used and the page dimensions.

See also: CONFIG

STOP (ST)

Syntax: STOP

Description: Stops sending characters from Protext's printer buffer to the printer, and displays the number of bytes left in the buffer. Use CONT to continue printing. The printer is deemed to be 'on STOP' after this command. Note that the printer will not stop immediately if the printer itself contains a buffer.

See also: ABANDON, BACK, CONT

SWAP (SW)

Syntax:

```
SWAP (<file number>)
SWAP (<filename>)
```

Description: Swaps between two documents in memory. All settings of the files and cursor, block markers etc are retained. See 'Multiple file editing' for full details.

SYMBOL (SYM)

Syntax:

```
SYMBOL <code> <n1> ... <n8>
SYMBOL <code> <n1> ... <n14>
SYMBOL <code> <n1> ... <n16>
```

Description: This command redefines the symbol that is shown on screen for a particular character. <code> is the ASCII code of the character, and is followed by 8, 14 or 16 single byte numbers defining the character matrix, starting at the top row and moving down. The number of parameters depends on the mode being used, as follows:

PC	*8x8*	*CGA graphics mode, EGA/VGA/MCGA/Hercules small characters (for sub/superscript and 43/50 line mode).*
	14x8	*EGA text and graphics modes.*
	16x8	*VGA/MCGA text and graphics mode.*
ST	*8x8*	*Medium res., High res. small characters (for subscript, superscript and 50 line mode).*
	16x8	*High resolution mode.*
Amiga	*8x8*	*Standard mode.*
	16x8	*Enhanced mode (with Enhanced Chip Set).*

Examples:

```
SYMBOL 32 0 0 0 0 0 0 85 170
SYMBOL 32 0 0 0 0 0 0 0 0 0 0 0 0 0 0 85 170
```

PC *Protext must be run in graphics mode or EGA/VGA/MCGA text mode in order to redefine characters.*

TAB

Syntax: TAB <column(s)>

Description: This command sets the tab stops which are used in program mode. A list of column positions may be given, separated by commas. The last number in the list may be preceded by '*'. This causes tabs to be set at equal intervals up to column 128.
TAB with no parameters sets the default tab positions (at every 8th column).

Example: TAB 8,15,*5 sets tabs at 8,15,20,25,30,...

Note: TAB *4 is a useful setting for many programming languages, including C and Pascal. This may be configured using CONFIG.

See also: PROG

TIME *ST only*

Syntax: *TIME <hh:mm:ss>*

Description: *Sets the system clock to the specified time.*

See also: *DATE*

PC *Use the MS-DOS command 'TIME'.*

Amiga *Use the AmigaDOS command 'DATE'.*

Archimedes *Use the 'Alarm' program from the Desktop.*

TOUCH *ST only*

Syntax: *TOUCH (dr:)(path\)(filename)*

Description: *Sets the date and time of the specified file or files to be the current time.*

See also: *UPDATE*

TYPE (T)

Syntax: TYPE <filename>

Description: Used to 'type' the contents of a Protext or ASCII file to the screen. This command displays Protext codes correctly and so is preferable to similar commands provided by the operating system.

The file is not loaded into memory, merely the contents displayed on the screen. All stored commands and rulers will be displayed and not acted on. This can provide a convenient means of viewing the contents of a file without loading it into memory. Whilst the file is being typed pressing ESC will pause the display. Pressing ESC a second time will cancel the command and any other key will resume.

Note:	In some respects TYPE may be thought of as duplicating the PRINTS command, but this is not really true, the difference being that TYPE just lists the contents of a file on the screen, whereas PRINTS displays on the screen what would be sent to the printer. TYPE will display stored commands.
See also:	PRINTS

TYPEWRITER (TW)

Syntax:	TW
Description:	This command puts the computer into direct typing mode. Everything that is typed is printed as soon as the RETURN key is pressed. Normal edit mode commands may be used to edit the text prior to pressing RETURN. Pressing ESC returns to Protext command mode. There are a number of possible uses for this command, including the easy addressing of envelopes, or adding postscripts to the end of letters. If Near Letter Quality or any font other than normal draft mode is required, the printer must be set up correctly before entering TW. If a document has just been printed in NLQ, then TW will continue in NLQ. Alternatively, many printers have methods of easily switching into NLQ.
Extensions:	A number equivalent to the number of spaces that the left margin should be inset.
Syntax:	TW <number>
Note:	This command defaults to zero margin, meaning that if text is normally printed with a side margin of 5, then it will be necessary to specify 5 when entering TW, to maintain the same left margin.
See also:	PRINT, PRINTON

UPDATE (UP) *ST / Amiga only*

Syntax: *UPDATE <sourcename> (destname)*

Description: *The same as COPY, except that if the destination filename already exists, the file is only copied if the source file has a later time.*

WINDOW (WIN)

Syntax: WINDOW <number>

Description: If a number is given the screen is split and that number of lines is allocated to the bottom window. If no number is given and the screen is not already split, it is split equally between top and bottom. If no number is given and the screen is already split then it is unsplit.

0 *Amiga / Archimedes only*
Syntax: *0*

Description: *Selects drive 0 (Amiga: df0:).*

1 *Amiga / Archimedes only*
Syntax: *1*

Description: *Selects drive 1 (Amiga : df1:), if fitted.*

2 *Amiga / Archimedes only*
Syntax: *2*

Description: *Selects drive 2 (Amiga : df2:), if fitted.*

3,4,5,6,7 *Archimedes only*
Syntax: *3*

Description: *Selects drive 3, 4, 5, 6 or 7 if fitted.*

24. Stored Commands Overview

Many of the features of Protext are called into operation by stored commands, so it is essential that their use is understood before some of the advanced features described in later chapters are used. This chapter covers some general topics, while the following chapter lists all stored commands.

Protext permits the controlling of various of the printing parameters in two ways, either through the use of the CONFIG command, or through the use of stored commands. CONFIG has the effect of changing the default settings of the various print options that are provided by Protext.

Stored commands can be used in Protext for many things, including setting page length, margin sizes, page numbering, printing text at the top or bottom of every page and line spacing. By using stored commands within a document, it is possible to set the appropriate options relating to that individual document, and these will take priority over the default settings. This is a very flexible method, since different types of document may require quite different margin sizes, page lengths or line spacing, for example.

The ruler line that is used to define the width of the text and tab settings is a special example of a stored command. This can be used in conjunction with the stored command '>SM' (side margin) to ensure that text is printed centrally on the paper.

When printing is taking place, the program will make use of the default settings for the various printing options set by CONFIG until a stored command is found which redefines that parameter, in which case the stored command has the effect of overriding the defaults, until either printing is complete or the stored command is redefined again by another stored command. It should be noted that some of these commands take immediate effect, whilst others only take effect from the next page printed.

A stored command line is denoted by a greater-than sign ('>') in the first column. Columns 2 and 3 contain a two letter code defining the instruction. The remainder of the line may contain a parameter for the command. A semicolon may be used to terminate the command, allowing a comment to be put on the same line. Any characters after a semicolon are ignored (unless the semicolon is enclosed in quotes). Any line with '>' in column 1 that is not followed by a known command is taken to be a ruler line if columns 2 and 3 each contain one of the characters: 'L', '-', '!' or a space.

Whenever text is printed, Protext performs a syntax check on the stored commands. If an unrecognised command is found, the syntax check stops and the error message 'Unknown stored command' is displayed. ESC should then be pressed to return to edit mode to correct the mistake. The cursor will be positioned on the line containing the unknown stored command.

Template Files

An important concept is that of creating a number of 'template' files, arranged for commonly used document formats, which are saved on disc and loaded when required. As an example, one template file might be for 'letter format', another for 'double spaced manuscript' and so on.

A template file would contain, amongst other things, a ruler line and a number of stored commands defining the page length, the margins, continuous or single sheet stationery, etc. In some cases the template might contain some standard text. For example, an invoice template could be a complete invoice with gaps to enter the name and address, details and the figures, or a letter template could contain the sender's address and telephone number.

When a new document is started, it is only necessary to load the appropriate template file and everything will automatically be configured to suit it. Another advantage of using stored commands is that the template will be saved with the document, ensuring that when the document is next loaded, all the parameters will be as before.

Example of a 'letter template'.

```
>CO Letter template
>SM 10          ; - Side margin 10
>PL 72          ; - Page length 72 (for 12 inch paper)
>CP OFF         ; - Continuous printing off (single sheet)
>----!---------------------------------------------------R
```

Example of a 'manuscript template'

```
>CO Manuscript template
>TM 2         ; - Top margin 2
>LS 2         ; - Double line spacing
>CP ON        ; - Continuous printing
>PN 1         ; - Page number from 1
>HE           TITLE OF DOCUMENT
>FO                    Page %
>--------!--------------!------------------------------R
```

Note: On occasions a document may contain a considerable number of stored commands. One effect of this, which may be noticed, is that scrolling through a document may be considerably slower, as Protext checks every stored command when a line is scrolled. If the CTRL-P option is used to disable page mode, the speed of scrolling will be increased.

Inter paragraph space

The stored command '>IP ON' causes blank lines between paragraphs to be printed as half lines. Thus a single line space between paragraphs gives 1½ line spacing. If two blank lines are left between paragraphs, double line spacing will be obtained. Three blank lines gives 2½ line spacing. '>IP OFF' cancels this and resumes normal line spacing.

Tab leader

Tab characters in the text are usually expanded to spaces when printed. The '>TL' command allows a different character to be specified. All tabs will then be expanded to this tab leader character except for leading tabs at the start of a line, which are always expanded to spaces.

Application example: To pad out a contents listing with dots between the text and the page numbers, use:

```
>TL .          ; before the contents
>TL ' '        ; after the contents
```

Headers, footers and page numbers

By default, no headers or footers are printed. A header or footer definition command is required. There are six commands which may be used to define header and footer text and each of them takes a string of text as the parameter. This is the string of text that will be printed at the top or bottom of the pages, such as a title, or a page number. A single space immediately following the command name is not included in the header or footer text to be printed.

'HE' and 'FO' are the two commands to define header and footer text which is to be printed on all pages. Sometimes it is necessary to print different text on the even numbered pages from the odd numbered ones and if so, '>EH' and '>EF' will define the text for even pages and '>OH' and '>OF' will define text for odd pages. With all these commands, which are described fully in the next chapter, the act of defining the text also has the effect of turning the headers or footers on.

Sometimes it will be necessary to turn headers and footers on or off at some stage in the printing. This can be done with the two stored commands, '>HE ON/OFF' and '>FO ON/OFF'. Turning headers or footers off has no effect on the header or footer text, other than the fact that it will not be printed, and they may be turned on again at any time. There is no restriction on the number of different headers and footers that may be used in a document and they may be changed at any time by using the appropriate header or footer definition command to define the new text.

Note: Header and footer commands must be inserted into the text before the start of the page to which they are to apply as they will only take effect when the next new page is printed.

Note: Since the first four columns are occupied by the stored command name, the text will be printed four columns to the left of where it appears on the screen. In order to position the header or footer correctly, relative to the rest of the text, the string should be entered before the stored command is inserted into the left hand columns. Similarly, if the text is to be centred, this should be done first, with CTRL-C and then move the cursor to column 1 and insert the appropriate stored command. To centre a proportional font header, a centre tab should be used.

Page numbers can be printed anywhere in the header or footer line. This is done by typing a percent symbol ('%'). The page number will then be printed starting in the column occupied by the percent symbol in the text. To print a percent symbol in a header of footer, type two percents ('%%').

By making use of odd and even headers or footers it is therefore possible to have page numbers printed on the outside edge of pages and different text on odd and even pages as well, in a form that would frequently be used for a book.

If headers or footers are turned on with the '>HE ON' or '>FO ON' commands, centred page numbers will automatically be printed unless alternative text is given with the header or footer commands. Using the command with no text will suppress the page number.

'>LF OFF' may be used to turn off the footer for the last page to be printed. This will not work if the document ends with a '>PA' command. One particular use for this command is when the footer contains the text, 'more' and the message would not be required on the last page.

Multiple line headers and footers

The header and footer commands can also be used to create headers or footers which consists of more than one line. Up to 9 lines may be used in any header or footer. The extra lines of text are specified by typing a number (2, 3, ...9) immediately following the stored command (HE, FO, EH, EF, OH, or OF). A single space after the number is not included in the header or footer text.

For example, the following commands will define a 3 line header:

```
>HM 4
>HE                    Protext Word Processor
>HE2
>EH3 WORD 2-%
>OH3                                                   WORD 2-%
```

Note: The header margin must contain at least as many lines as there are lines of text in the header. If the margin is too small any extra header lines will not be printed. The header text will always be printed in the topmost lines of the header margin, and the footer text in the bottom lines of the footer margin.

Pagination

Pagination is controlled by many factors. Protext is supplied with a set of default values which are suited to use with either A4 single sheet paper, or continuous 11 inch paper, but any of the parameters may be altered by use of stored commands. The values of all the margins may be changed, as well as page lengths and line spacing. Protext automatically calculates the number of lines that will actually be printed on a page. There is no need to print a page to see where the page breaks will occur, as Protext displays page break lines while editing. The page and line numbers are shown on the status lines at the top of the screen. The line number shown is always the number of the line as it will actually be printed, having made any allowances necessary for headers and footers.

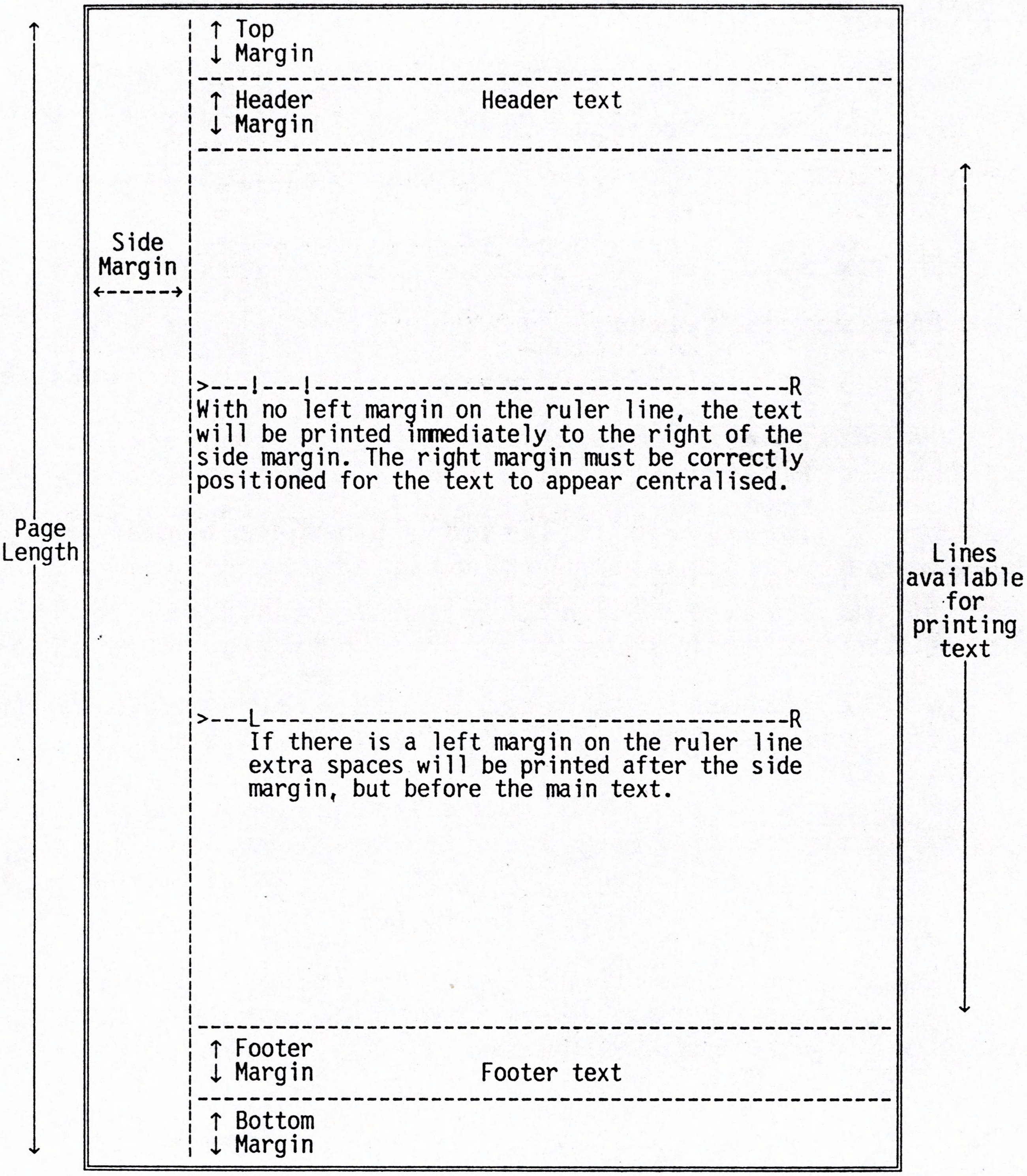
Example of setting margins with stored commands:

```
>PL 72        ; the physical paper size 12" paper at 6 lines per inch
>TM 0         ; no top margin
>FM 4         ; footer margin 4
>FO ON        ; so the footer margin is used
>BM 8         ; bottom margin 8
```

The number of lines actually printed on a page with this example will be 60, that is the page length of 72 less the footer and bottom margin sizes. The header margin is zero since headers have not been turned on. See the next chapter for details of all these commands.

By making use of the editing commands, CTRL-(and CTRL-) to move forward and backwards a page at a time it is very easy to see where page breaks are going to occur. If the position in the text is unsuitable, a page break can be forced at any time by inserting the '>PA' stored command into the text at a more suitable location. Alternatively '>OP' and '>EP' can be used to ensure that the next page to be printed has an even or an odd number.

The stored command '>SM' is used to determine how many spaces are printed at the start of every line, to ensure that the text is printed in the correct position across the paper. In addition, it is possible to specify different side margins for odd and even pages, using the '>EM' and '>OM' commands. This can be useful to ensure that a larger margin is left on odd numbered pages and a smaller one on even numbered pages, so that text is not printed too near to the fold of a book.


↑ Top
↓ Margin
↑ Header
↓ Margin
Header text
Side
Margin
←------→
Page
Length
Lines
available
for
printing
text
↑ Footer
↓ Margin
Footer text
↑ Bottom
↓ Margin

```
>---!---!--------------------------------------R
With no left margin on the ruler line, the text
will be printed immediately to the right of the
side margin. The right margin must be correctly
positioned for the text to appear centralised.
```

```
>---L------------------------------------------R
    If there is a left margin on the ruler line
    extra spaces will be printed after the side
    margin, but before the main text.
```

Note: The page length is the size of the paper. If a printer is unable to print on part of a sheet (for example the top few lines on single sheets), these lines must be included within the top or bottom margins. The number of lines of text actually printed will be the page length less the total of the four margins (top, header, footer, bottom).
Note: PRINTS will show on screen how the printed page will appear.

Footnotes and Endnotes

There are two things to be done to create a footnote: (i) define the footnote text, (ii) insert a marker at the point in the text to which the footnote refers.

There are two forms of the '>FN' command for defining footnote text.

```
>FN <text>
```

defines a single line footnote[1].

```
>FN
... any number of lines of text
>FE
```

defines a multi-line footnote[2]. The text of a multi-line footnote will be formatted in the same way as normal text, and any printer control codes may be used within the footnote as long as matching off codes are included at the end of the note. Stored commands may not be used within a footnote.

The '>FE' command is used to mark the end of a footnote. This may be omitted if a footnote is followed immediately by another '>FN' command.

Any number of footnotes may be entered. These are simply held by Protext as a list and may appear anywhere before the corresponding footnote marker. For example, it is possible to put all the footnotes at the start of a document, or they could be entered at the relevant point in the text.

A footnote marker is entered into the text with CTRL-@ F. This will appear as a superscript n in the text (^{n}). When printed it will appear as a superscript number as seen in the examples on this page.

---

[1] This is an example of a single line footnote.

[2] This is an example of a footnote that takes up three lines and is formatted and right justified. It will also be noted that the footnote is printed in a mixture of two proportional fonts of different sizes and is correctly justified.

The footnote text is not related to the marker when editing. When the text is printed and a footnote marker is reached in the text, Protext will print the next reference number and use the next footnote text line. If there is insufficient space on a page for the footnote (for example if a footnote marker is placed on the last line of a page) the text will be carried forward to the next page. Footnote numbering restarts at 1 at the start of each page unless a note has been carried forward.

Note: the footnote text must be defined before the corresponding marker is used.

Endnotes

Endnotes work in the same way as footnotes except that whereas footnotes are printed at the foot of the page, endnotes are printed at the end of the document (or at any required point such as the end of a chapter).

The same commands ('>FN', '>FE') and the same footnote markers are used for both footnotes and endnotes. To select endnotes instead of footnotes use the '>EN' command.

```
>EN ON          ; use endnotes from here
>EN OFF         ; revert to using footnotes
```

Endnotes will normally be printed at the end of the document. If it is necessary to print the endnotes earlier, such as at the end of a chapter in a document of several chapters, use the command '>DE' (dump endnotes) at the end of the last page of the chapter.

```
>DE     ; print the endnotes immediately
```

25. Stored Commands

This chapter presents an alphabetical list of all stored commands, with full syntax descriptions of all the stored commands and default values, where applicable. Note that many of the default values may be changed by the use of the CONFIG program. The key to the syntax descriptions is given at the end of the chapter.

Note: Comments may be placed on the same line as any stored command and can serve as a reminder of the function of the command. The semi-colon (;) is used to indicate to Protext that any further text is merely a comment and is not to be acted on.

The Stored Commands

AV	Ask for Variable from keyboard.
Syntax:	>AV (<string>) <identifier> (<int>) { (<string>) <identifier> (<int>) }
Examples:	>AV "Name",name,20,"Address line 1",addr1 >AV name,addr1,addr2,addr3,addr4
Description:	Takes information typed on keyboard and assigns to variable. See 'Mail Merging'.

BM	Bottom Margin.
Syntax:	>BM <int>
Example:	>BM 10
Description:	The bottom margin is the number of blank lines left at the bottom of every page, below the footer. The default is 3.

CC	Redefine Control Code.
Syntax:	>CC <letter> {<code>} (; {<code>})
Examples:	>CC "h",27,82,6 ; 27,82,3 >CC @
Description:	Any printer control code (@, a-z) may be defined. The first item after 'CC' is the control code letter, followed by the sequence of codes for 'on', followed (optionally) by a semicolon and then the codes for 'off'. The last example stops the reset code being sent. See also 'Creating Printer Drivers'.
CE	CEntre text in line.
Syntax:	>CE <text>
Description:	This command is used to centre the line during re-formatting and during printing.
CF	Close File.
Syntax:	>CF
Description:	Closes data file. See 'Mail Merging'.
CO	COmment.
Syntax:	>CO <message>
Examples:	>CO These lines do nothing except permit you >>> to put comments and reminders in the text.
Description:	This command allows the entry of notes about the document and may be used anywhere in the text. When printing is taking place all comment lines are ignored. '>>>' is also permitted as an alternative to '>CO'.
Note:	It is also possible to put comments at the end of any stored command lines, by preceding them with a semi colon (;). Anything after the semi colon will be ignored.
Note:	Comment lines are useful for putting in index references when the text of the index entry does not appear in the text.

CP ON/OFF	Continuous/Single sheet Printing.
Syntax:	>CP ON >CP OFF
Description:	'ON' selects continuous printing, 'OFF' selects single sheet printing. The default setting is 'ON'.

CS	Clear Screen.
Syntax:	>CS <message>
Example:	>CS Enter Information when requested.
Description:	The screen is cleared and any message is then displayed on the screen.

CW	Define Character Width
Syntax:	>CW <int>
Examples:	
Description:	This command is not needed and is supported only for backwards compatibility.

DC	Set the Decimal Character
Syntax:	>DC <character>
Examples:	>DC , >DC "."
Description:	Defines the character used as a decimal point. Comma and full stop are the only valid characters. The default is full stop. CONFIG would normally be used to set the decimal character.

DE	Dump Endnotes
Syntax:	>DE
Description:	Cause all pending endnotes to be printed immediately. Most commonly used at the end of a chapter.

DF	**D**efine data **F**ile or files.
Syntax:	>DF <filename> {<filename>}
Examples:	>DF "data" >DF names1,names2,names3
Description:	Opens data file for mail merging. See 'Mail Merging'.
Note:	It is not possible to have more than one DF command in use at a time. If more than one DF command is required, a CF command must close the first one before the next is opened.

DM	**D**isplay **M**essage.
Syntax:	>DM <message>
Example:	>DM Daisy wheel will soon require changing to Courier 10
Description:	The message is displayed on the screen.

DP	Set number of **D**ecimal **P**laces
Syntax:	>DP <integer> >DP r
Description:	Sets the number of decimal places shown when numeric variables are displayed. The default is 2. '>DP r' or '>DP R' is used to shown variables in Roman numerals. See 'Mail Merging'.

DU	**DU**mp values of currently defined variables
Syntax:	>DU
Description:	A list of all defined variables, together with their values and precision (defined with >DP) is displayed on the screen. This is primarily for the purpose of debugging mail merge applications.

EA	**E**nd printing **A**t page.
Syntax:	>EA <integer>
Example:	>EA 47
Description:	Sets the last page number to be printed. The default is the last page of the document. This command is ignored if the print dialogue box or PP/PPQ/PPS with a list of pages is used.

EC	**End Column**
Syntax:	>EC
Description:	When printing multiple columns, terminates a column and causes printing to resume at the top of the next column. See '>PC'.

EF	**Even Footer text definition.**
Syntax:	>EF <message> >EFn <message>
Example:	>EF This is printed at the foot of even pages only
Description:	Permits the printing of the specified text at the foot of even numbered pages only. Page numbers may be included and more than one line of text may be used (see the section on Headers and Footers in the previous chapter).

EH	**Even Header text definition.**
Syntax:	>EH <message> >EHn <message>
Example:	>EH This is printed at the top of even pages only
Description:	Permits the printing of the specified text at the top of even numbered pages only. Page numbers may be included and more than one line of text may be used (see the section on Headers and Footers in the previous chapter).

EI	**EndIf.**
Syntax:	>EI
Description:	Ends IF block, matches with ID, IE, IF or IU. See 'Mail Merging'.

EL	**ELse.**
Syntax:	>EL
Description:	Starts EL block, matches with ID, IE, IF or IU. IF blocks may be nested within EL blocks. See 'Mail Merging'.

EM	**E**ven **M**argin.
Syntax:	>EM <int>
Example:	>EM 3
Description:	Defines the number of spaces printed at the left hand edge of every even numbered page. The default is 5.

EN ON/OFF	Turn **E**nd**N**otes on and off
Syntax:	>EN ON >EN OFF
Description:	When enabled all footnote definitions (using '>FN') are treated as endnotes and printed at the end of the document or the next '>DE' command.

EP	**E**ven **P**age throw.
Syntax:	>EP (<int>)
Example:	>EP 10
Description:	Similar to the normal page throw command '>PA' except that if the new page to be printed has an odd number, a complete blank page will be printed. The text is therefore made to start at the top of an even numbered page. If a parameter is given, a new page will be started only if the number of blank lines remaining on the current page is less than or equal to the number specified.

EX	**EX**ternal command.
Syntax:	>EX <string> >EX ?<string>
Example:	>EX SERIAL
Description:	This permits commands that are normally used from command mode to be executed during the printing of the text. This facility is provided for flexibility. In the above example it is being used to change the printer port to the serial port prior to loading a new printer driver with '>PR'. (*Example not applicable to Amiga*).

>EX may be followed by a ? to specify that the command should only be executed if the line is in a block being printed. Otherwise all >EX commands are executed even if printing a block and the >EX line lies outside that block.

For example you may have a file to be copied directly to the printer from within the text. This may be done by:

>EX ?COPY PICTURE *(example for PC only)*

FD	Define Font Default
Syntax:	>FD <string of control code letters>
Description:	Explicitly defines the control code or combination of control codes that will be used as the default font for the purposes of determining the ruler and tab widths and margin sizes. Usually only used at the start of a document.
FE	Footnote End
Syntax:	>FE
Description:	Terminates a multi-line footnote definition.
FF ON/OFF	Form Feeds enabled/disabled.
Syntax:	>FF ON >FF OFF
Description:	When enabled a form feed character (12) is printed at the end of each page. The default setting is 'OFF'.
Note:	This command is often useful with laser printers and sheet feeders. If '>FF ON' is to be used, it is important to make sure that the correct page length is set in the printer, if necessary using the 'OC' command to send the correct page length codes to the printer.

FM	Footer **M**argin.
Syntax:	`>FM <int>`
Example:	`>FM 3`
Description:	Defines the number of lines between the last line of text and the bottom margin. The lines in the footer margin may have footer text printed on them. The default is 2 if footers are on, otherwise zero.

FN	Define **F**oot**N**ote text
Syntax:	`>FN <text>        ; defines a single line note`
	`>FN               ; starts a multi-line definition`
Example:	`>FN this is a footnote`
Description:	Defines the text for a footnote. A footnote marker must be inserted into the text to cause the footnote to be printed. The '>FE' command is used to terminate a multi-line footnote definition. (See the section on Footnotes in the previous chapter).

FO	**FO**oter text definition.
Syntax:	`>FO  <message>`
	`>FOn <message>`
Example:	`>FO        Continued on next page`
	`>FO2                 %`
Description:	Defines the footer text which is to be printed at the bottom of every page. The text is printed on the bottom line of the footer margin. Page numbers may be included and more than one line of text may be used (see the section on Headers and Footers in the previous chapter).

FO ON/OFF Turn **FO**oters on or off, without affecting the footer text

Syntax:
>FO ON
>FO OFF

Description: Used to determine whether the footer text is to be printed or not. It does not affect the footer text contents.

Note: The text to be printed must have been previously defined with the >FO 'text' command.

FP ON/OFF **F**ormatting whilst **P**rinting on/off.

Syntax:
>FP ON
>FP OFF

Description: When enabled, the text is formatted during the printing operation. The command '>RJ' controls whether right justification is carried out or not. '>FP' should always be used in conjunction with '>RJ ON/OFF', but in the event that '>RJ' is not specified, formatting of the document will take place according to the current setting of edit mode justification when the file is printed.
Even when disabled, any paragraph that includes a variable reference will be re-formatted as a matter of course (see 'Mail merging'). The default setting would be 'OFF', though most printer drivers set 'PP ON' which changes the default to 'ON'.

FX Fi**X** text.

Syntax:
>FX ON
>FX OFF

Description: Using '>FX ON' ensures that text is not reformatted during a FORMAT command or when formatting takes place during printing or editing.
'FX OFF' will resume formatting of the text from that point. This can be very useful where sections of text are complex and formatting might destroy the appearance, for example when text has been formatted in columns using box mode. The default setting is 'OFF'.

Note: >FX differs from the FIX command in that the section of text can still be edited and reformatted manually and by removing the '>FX' commands, it will revert to normal.

GC	Generate Contents
Syntax:	>GC <filename> <options>
Description:	Defines options for table of contents generation. See 'Index and contents'.
GI	Generate Index
Syntax:	>GI <filename>
Description:	Defines options for index generation. See 'Index and contents'.
HE	**HE**ader text definition.
Syntax:	>HE <message> >HEn <message>
Examples:	>HE Chapter 1 >HE2 ____
Description:	Turns headers on and defines the header text which is to be printed at the top of every page. The text is printed on the top line of the header margin. Page numbers may be included and more than one line of text may be used (see the section on Headers and Footers in the previous chapter).
HE ON/OFF	Turn **HE**aders on or off without affecting the text.
Syntax:	>HE ON >HE OFF
Description:	Used to determine whether the header text is to be printed or not. It does not affect the header text contents.
Note:	The text to be printed must have been previously defined with the >HE 'text' command.

HM	**Header Margin.**
Syntax:	>HM <int>
Example:	>HM 4
Description:	Defines the number of lines between the top margin and the first line of the text. The lines in the header margin may have header text printed on them. The default is 2 if headers are on, otherwise zero.
IB	**Insert Binary file**
Syntax:	>IB <filename> >IB ?<filename>
Description:	Reads a file and copies it byte for byte to the output stream. If the filename is prefaced with a question mark the operation is only carried out when the command occurs in a section of text that is being printed. Uses for this command include printing graphics in the middle of a document and downloading fonts to a printer.
ID	**If Defined.**
Syntax:	>ID <identifier>
Description:	The following block is printed if the variable is defined. See 'Mail Merging'.
IE	**If Exhausted.**
Syntax:	>IE
Description:	Tests data file for no more data and if so, block is printed. Matches with EI. See 'Mail Merging'.
IF	**IF** condition true - print.
Syntax:	>IF <condition>
Description:	The following block is printed if the condition is true. IF blocks may be nested to a depth of 7. Matches with EI. See 'Mail Merging'.

IN INsert file into text.
Syntax: >IN filename

Description: This command may be used at any point in the text of a document and the named file is read from disc and the contents printed. When the end of the file is reached, printing of the original file continues at the line after the IN command. The file is not loaded into memory, merely printed. Any number of 'IN' commands may be issued.

Note: Drive letters and paths may be specified preceding the filename, in the normal fashion.

Another use of 'IN' is to permit the continuous printing of a number of files as one document. A text file consisting of the following entries will, when printed, result in 'file1' being printed, followed by 'file2' and finally 'file3'.

```
>IN file1
>IN file2
>IN file3
```

One of the advantages of this is that page numbering, headers, footers, or margins set at the start of the document will follow through correctly, unless altered by any stored commands in the files being printed. In this way, it would be possible to print out a whole book, from beginning to end, in one run, with headings and page numbers running correctly through.

IP Turn **I**nter-**P**aragraph space increase on and off.
Syntax: >IP ON
>IP OFF

Description: When enabled blank lines between paragraphs are printed as half lines.

IU **I**f **U**ndefined.
Syntax: >IU <identifier>

Description: The following block is printed if the variable is not defined. See 'Mail Merging'.

LF	**Last page Footer.**
Syntax:	>LF ON >LF OFF
Description:	This command may be used to ensure that any footer text is not printed on the last page. The default is 'ON', which means that any footer text will be printed on the last page if footers are turned on. This command is useful, as it allows you to turn off the footer for the last page only, with a command which may be a part of a template file. For example, footers might be defined to print 'more' at the bottom of every page. Obviously the 'more' message would not be wanted on the final page. If '>LF OFF' is used in the text prior to the start of the last page, then the footer will not be printed. Without this command, it is necessary to find the start of the last page and insert a stored command to turn footers off before the start of the page.
Note:	If this command is used, then it is important not to use the 'PA', 'EP' or 'OP' commands at the end of the document, otherwise Protext will think there is more text to follow and print the footer.

LS	**Line Spacing.**
Syntax:	>LS <int>
Examples:	>LS 2 >LS 1.5 >LS 2½
Description:	Defines the number of line feeds to be printed after each line of text. Half line feeds are permitted as shown in the examples, but the appropriate codes must be specified in the printer driver. The default is 1, that is single spacing.
Note:	Do not use >LS 0 to prevent double line feeds - this would work but would have the side effect of printing the whole text without any page breaks. Unwanted double line feeds should be corrected by altering the dip switch settings on the printer (see 'Hints and Tips').

MC Define **M**icrospace **C**ode sequence
Syntax: `>MC <code> {<code>}`
Example: `>MC 27 "L" 1 0 0`

Description: This defines the sequence of codes that moves the print head by the smallest possible amount. This will normally be defined in the printer driver and not by the stored command. See 'Creating Printer Drivers'.

MF Define **M**icrospacing **F**actor
Syntax: `>MF <int>`
Example: `>MF 4`

Description: Defines microspacing factor. The default value is 1. This would normally be defined within the printer driver, not by the stored command. See 'Creating Printer Drivers'.

MM Microspacing **M**ethod selection.
Syntax: `>MM <int>`
Examples:

```
>MM 2        ; use for HP LaserJet and compatible printers
>MM 4        ; use for Epson compatible 24 pin printers
```

Description: MM is used to select the method used for proportional printing and microspacing. This would normally be defined within the printer driver, not by the stored command. See Creating Printer Drivers'.

MP Turns **M**ultiple **P**rinting on or off
Syntax:

```
>MP ON
>MP OFF
```

Description: Multiple printing is used on laser printer printers to print several copies of a document quickly, only sending it once to the printer. This command may be useful occasionally to turn multiple printing off in order to avoid the need to collate the document after printing.

MS	Turns MicroSpacing on or off
Syntax:	>MS ON
	>MS OFF
Description:	When microspacing is on, Protext will distribute the extra space on each line (soft spaces) evenly between the words. This gives a better appearance to the text, though inferior to proportionally printed text. Microspacing has the advantage that it can be used on non-proportional printers and can be used if a printer driver containing character widths is not available.

NC	Number of Copies.
Syntax:	>NC <int>
Example:	>NC 3
Description:	Sets the number of copies to be printed. The default is 1.
Note:	This command is ignored if a data file is defined.

NP ON/OFF	Enable/disable New Page at end of printing.
Syntax:	>NP ON
	>NP OFF
Description:	When enabled the paper is moved to the top of the next page at the end of printing the document. When disabled the paper is stopped as soon as the last line of text has been printed. The default setting is 'ON'.
Note:	It may be useful to turn this setting off, in order to add 'personal' post scripts to letters, by using Typewriter mode (See 'Commands').

OC Output Codes.

Syntax:

```
>OC <code> {<code>}
```

Examples:

```
>OC 27,"3",12          ; change line spacing
>OC 27,J,20
>OC 27,"l",5           ; set left margin (use >SM 0 as well)
```

Description: This command takes a list of codes which are sent directly to the printer. It is particularly useful for defining character matrices on the printer when a long string of codes are required. The printer driver could be used, but can only hold a limited number of characters.

The codes may be sent as decimal numbers, hex numbers (prefix with & or #) or ASCII characters. If an item is not a number it is sent as an ASCII character. To send the ASCII code of a decimal digit enclose the digit in quotes. Codes may be separated by commas or spaces.

OF Odd Footer definition.

Syntax:

```
>OF  <message>
>OFn <message>
```

Example:

```
>OF  This is the bottom of an odd numbered page
```

Description: Defines the footer text which is to be printed at the bottom of every odd numbered page. Page numbers may be included and more than one line of text may be used (see the section on Headers and Footers in the previous chapter).

OH Odd Header definition.

Syntax:

```
>OH  <message>
>OHn <message>
```

Example:

```
>OH  This is the top of an odd numbered page
```

Description: Defines the header text which is to be printed at the top of every odd numbered page. Page numbers may be included and more than one line of text may be used (see the section on Headers and Footers in the previous chapter).

OM	**O**dd **M**argin.
Syntax:	>OM <int>
Example:	>OM 7
Description:	Defines the number of spaces printed at the left hand edge of every odd numbered page. The default is 5.

OP	**O**dd **P**age throw.
Syntax:	>OP (<int)>
Example:	>OP
Description:	Similar to the normal page throw command '>PA' except that if the new page to be printed has an even number, a complete blank page will be printed. The text is therefore made to start at the top of an odd numbered page. If a parameter is given, a new page will be started only if the number of blank lines remaining on the current page is less than or equal to the number specified.

PA	**PA**ge throw.
Syntax:	>PA (<int>)
Examples:	>PA >PA 6
Description:	In its simplest form (>PA), this command forces a new page to be started and printing of the text will continue from the top of the next page. If the PA command occurs at a natural page break then it will not force an extra new page. If a parameter is given (e.g. >PA 6) a new page will be started only if the number of blank lines remaining on the current page is less than or equal to the number specified.

PC	Print Columns
Syntax:	>PC <int> (<int>)
Description:	Define the number of columns for multiple column printing. If the second number is given this defines the total width of the printed text in columns, otherwise this defaults to the same as the currently active ruler. To revert to normal printing use the command '>PC 1'. See also '>EC'.

PE ON Print Even numbered pages only.
Syntax: >PE ON

Description: Only pages with even numbers will be printed when printing takes place. '>PE OFF' causes all pages to be printed. If only even numbered pages are printed the paper may be turned over at the end of a print run and the odd numbered pages printed on the reverse side, using the '>PO' command. The default is 'OFF'.

PL **Page Length.**
Syntax: >PL <int>
Example: >PL 72

Description: PL defines the actual size of a page of paper, and is different from the number of lines of text printed on a page. The default page length is 66, which is the correct setting for standard 11 inch paper.

Note: The number of lines of text printed on a page is given by the formula:-

PL-TM-BM(-HM if headers on)(-FM if footers on)

i.e. the page length minus the sum of the sizes of the margins.

PN **Page Number.**
Syntax: >PN <integer>
Example: >PN 1001

Description: The page number can be printed at the top or bottom of every page, or specified pages. (see stored commands EF, EH, FO, HE, OF, OH and FO). PN sets the page number, which can be set to any permitted value at any point in the text. The largest number allowed is 65535. If no PN command is used, the first page will be number 1 by default and numbers will increase by 1 for each new page.

PO ON Print **O**dd numbered pages only.
Syntax: >PO ON

Description: Only pages with odd numbers will be printed when printing takes place. '>PO OFF' causes all pages to be printed. The default is 'ON'.

PP **P**roportional **P**rinting on/off.
Syntax: >PP ON
>PP OFF

Description: This command turns proportional printing on or off. The default is 'OFF'. When in operation, Protext will automatically calculate the widths of each individual character and if there is room at the end of a line, it will check to see whether the first word on the next line will fit and if so, it will pull it up and use it. Often this command is unnecessary because many printer drivers default to proportional printing.

When proportional printing is being carried out in this fashion, the default font determines the width of the line. See 'Proportional printing' for details.

This command is primarily intended for use with a proportional daisywheel printer or laser printer. It can also be used with the proportional mode of a dot matrix printer, though in some cases this is not recommended as the speed of printing is dramatically reduced. It should not be used with a fixed pitch daisywheel or a non-proportional dot matrix printer.

PR Load **PR**inter driver.
Syntax: >PR <filename>
Example: >PR juki6100

Description: The named printer driver is loaded. The above example will cause Protext to look for a file called 'JUKI6100.PPD'. All the control code and character translations are set to the new values.

Note:	If the printer for which the new driver is being loaded is connected to a different output port from the existing printer, it will be necessary to change the port prior to using this command. This may be done with the '>EX' stored command. For example (*PC only*) to change the port to the second serial port (SER2), use '>EX SERIAL 2', followed by '>PR'.
PS ON/OFF	Paper Sensing on/off.
Syntax:	>PS ON >PS OFF
Description:	This command will only work with some printers. The effect of using it is to put the printer 'off line' at the end of a page, to allow changing of single sheet paper when background printing is in use. The section on 'Background printing' gives full details. The default setting is 'OFF'.
Note:	This command does not work with some printers on which paper sensing only works with continuous paper. An example is the Kaga Taxan.
PW	Proportional character Width.
Syntax:	>PW <integer>
Example:	>PW 12
Description:	This command may be used to specify the printed line width for proportional printing. This will normally be defined in the printer driver and not by the stored command. See 'Creating Printer Drivers'.
RC	Redefine Character.
Syntax:	>RC <character> = <code> {<code>}
Examples:	>RC £ 27 ">" 1 27 "=" >RC "!" = "!" 8 "."
Description:	The first item after 'RC' is the character to be redefined, this is followed by the sequence of codes to be printed whenever the character occurs. The codes may be entered in the same way as for the 'OC' command, above. If a character is redefined, the printer driver definition can be restored by an 'RC' command with no code sequence, e.g. '>RC !'.

RJ ON/OFF Right Justifying on/off.
Syntax: >RJ OFF
>RJ ON

Description: This command is used to select whether right justification is carried out, when the formatting whilst printing option '>FP' has been selected. The default setting is whatever the edit mode setting of 'justify' is (set with CTRL-J) when printing.

Note: Unless '>FP' is set to 'ON', this command will have no effect.

RP RePeat.
Syntax: >RP

Description: Start of repeat loop. Matches with UN. Repeat loops may not be nested. See 'Mail Merging'.

RU Read variable Unconditionally.
Syntax: >RU <identifier> {<identifier>}

Description: Read variables from data file. See 'Mail Merging'.

RV Read Variable from data file.
Syntax: >RV <identifier> {<identifier>}
Example: >RV name,addr1,addr2,addr3,addr4,addr5,dummy

Description: Read variables from data file, filling spare variables with blanks if a blank line is read. See 'Mail Merging'.

SA Start printing At page.
Syntax: >SA <integer>
Example: >SA 41

Description: Sets the first page number to be printed. The default is 1. This command is ignored if the print dialogue box or PP/PPQ/PPS with a list of pages is used.

SC Turn Spell Checking on or off for a section.

Syntax:
```
>SC ON
>SC OFF
```

Description: Used to disable edit mode spell checking (including spell checking whilst typing) for a section of text.

SK SKip copy if condition is true.

Syntax:
```
>SK <condition>
```

Description: Skips to start of document if condition is true. See 'Mail Merging'.

SL Define Spell checking Language

Syntax:
```
>SL <language number> <dictionary list>
```

Example:
```
>SL 7 arnorsve.qic användar.lex arnorsve.lex
```

Description: '>SL' can be used to change the spell checking language at any point in a document. It requires both the language and the dictionaries to be given and takes the following form:

```
>SL n <qic dict> <update dict> <list of 0-5 dicts>
```

The language numbers are the same as the keyboard language numbers (see the documentation on the 'LANG' command). If '>SL' is being used in a document, the first '>SL' line should be at the start of the document (before the first text line) so that the language and dictionaries are defined for the whole document.

SM	Side **M**argin.
Syntax:	`>SM <int>`
Example:	`>SM 10`
Description:	Defines the number of spaces printed at the start of every text line between the paper's edge and the first column of text. The default is 5. This is called 'side margin' to distinguish it from the left margin that can be set on a ruler line. The Protext default ruler and side margin have been chosen for this reason.
Note:	Setting the side margin is equivalent to setting both even and odd margins to the same value.
Note:	Use this command to print text centrally on the paper, and leave the left margin of the ruler set at column 1 (except where text is indented).

ST	**ST**op printing.
Syntax:	`>ST <message>`
Example:	`>ST    No further text to be printed.`
Description:	Printing stops immediately. Any message is displayed on the screen. Further text is not printed.

SV	**S**et **V**ariable.
Syntax:	`>SV <identifier> = <expression>` `{<identifier> = <expression>}`
Description:	Assigns value of expression or expressions to variable or variables. See 'Mail Merging'.

TL	Set **T**ab **L**eader character.
Syntax:	`>TL <character>`
Examples:	`>TL .` `>TL " "`
Description:	Changes the character to which tabs are expanded when printed.

TM	Top Margin.
Syntax:	`>TM <int>`
Example:	`>TM 0`
Description:	The top margin is the number of blank lines left at the top of every page, above the header. The default is 3.
UN	**UN**til.
Syntax:	`>UN <condition>`
Description:	Ends repeat loop. Repeats if condition is false. See 'Mail Merging'.
WC	The file being **W**ritten to is **C**losed.
Syntax:	`>WC`
Description:	See 'Mail Merging'.
WF	**W**riting to **F**ile - open file with filename.
Syntax:	`>WF <filename> (A)`
Examples:	`>WF newdata` `>WF olddata A`
Description:	A file of the specified name will be opened for writing. The '>WF ON/OFF' commands determine whether the text is written to the file or not, when printing takes place. See below and mail merging for full details. The A option causes data to be appended to the file. If no A is given a new file will be created.

WF ON/OFF	Turn on and off Writing to the File.
Syntax:	>WF ON >WF OFF
Description:	When printing takes place, if writing is turned on, all text is sent to the file, including stored commands, printer control codes, rulers etc. at the same time as the document is printed normally. The file when closed is saved as Protext document and can be edited and printed as a normal document. It is principally intended for use with mail merging documents, where it provides the ability to save individual copies of any documents printed. See mail merging for more details.
WM	Write Message to output file.
Syntax:	>WM <message>
Example:	>WM This message will be sent to the disc file.
Description:	This command is used in conjunction with the '>WF' commands and is used to send messages to the output file opened with the '>WF filename' command.
WT	WaiT.
Syntax:	>WT <message>
Example:	>WT Change Daisywheel now and press any key to continue.
Description:	Printing pauses and the message is displayed on the screen. Printing resumes when a key is pressed. Pressing ESC will abandon further printing and return to command mode.
ZM	Zero all Margins.
Syntax:	>ZM
Description:	Following this command all margins will be set to zero. This command has a number of uses, one in particular being when used in conjunction with SAVEA to create a pure Ascii file. If SAVEA is used without this command, all margins and page breaks will also be sent to the file.
Note:	Any margins required can be reset after the use of this command, using the appropriate stored command.

Key to Syntax

(...)	an item in parentheses is optional.
{...}	an item in curly brackets may occur any number of times (possibly zero).
< character >	an ASCII character, optionally in quotes.
< string >	a sequence of ASCII characters, enclosed in quotes.
< message >	a sequence of ASCII characters, optionally in quotes. It may include variable references.
< filename >	a legal file identifier under the active filing system.
< integer >	a number between 0 and 65535
< int >	a number between 0 and 255.
n	a number between 1 and 9.
< identifier >	a variable identifier. Must start with a letter and may contain letters, digits, ".", "_" and "?".
< code >	an ASCII code. The codes may be specified as decimal numbers, hex numbers (prefix with & or #) or ASCII characters. If an item is not a number it is sent as an ASCII character. To send the ASCII code of a decimal digit enclose the digit in quotes. Codes may be separated by commas or spaces.

< expression >	is < num expr > or < str expr >
< num expr >	is an expression with a numeric result (see below)
< str expr >	is an expression with a string result
< condition >	is an expression with a numeric result, typically comparing two numbers or strings. A non zero result indicates 'true'.

separators	Where more than one parameter is listed on a command line the parameters may be separated by spaces, commas, or equals signs. This is not shown in the syntax descriptions.

Variable references:

To print the value of a variable at any point in the text, type the variable name with an "&" immediately before and after it, e.g. &name&. If the value of the variable is a null string, and there is nothing else on the line, then the line is omitted altogether. If the value of the variable is the null string and there is a space after the variable reference then the space is not printed.
If the blank line or space is required, then type "!" before and after the variable name, e.g. !name!.

Precision:	When a value is assigned to a variable the value is rounded to the precision of that variable (determined by the '>DP' value in effect when the variable was first defined).

The Expression Evaluator

Protext incorporates a sophisticated expression evaluator which is capable of evaluating complex expressions. It is also capable of carrying out extensive text manipulation processes.

Expression evaluation operators

The following is a list of all operators which may be used in Protext expressions and is listed according to the priority in which they are evaluated.

Note:
- Curved brackets may be used to force the priority of parts of an expression.
- Where more than one operator of the same priority is found, evaluation takes place from left to right.
- The case of operators is unimportant. 'DIV' is the same as 'div' or 'Div' for this purpose.
- In some cases more than one symbol or word may be provided for the same operator. This is for convenience only. '=' and '==' both mean 'equal to', for example.

In the following list, the entries under 'parm type' and 'result type' describe the type of 'item' expected and returned by the operation. In the case of 'parm type', two characters indicate the type of each 'item' required, before and after the operator.

n	Numeric
p	Polymorphic - either String or Numeric
s	String

Operators

	symbol or word	parm types	result type	description
1.				
	(			open bracket
2. Unary Operator				
	+	n	n	unary plus
	-	n	n	unary minus
	!	n	n	logical negation
	~	n	n	logical negation
	abs	n	n	absolute value
	datenum	s	n	convert date to number for calculation
	dayofweek	s	s	return name of day for given date
	frac	n	n	fractional part of
	int	n	n	integer part of
	isdate	s	n	test whether string is a valid date
	isnum	s	n	test whether string is a valid number
	ldate	n	s	convert number to long date (28 July 1989)
	len	s	n	length of string
	lwr	s	s	convert string to lower case
	not	n	n	logical negation
	sdate	n	s	convert number to short date (28/07/89)
	upr	s	s	convert string to upper case
3.				
	*	nn	n	multiplication
	/	nn	n	division
	div	nn	n	integer division
	mod	nn	n	remainder from integer division
4.				
	+	pp	n/s	addition or string concatenation
	-	nn	n	subtraction
	$+	ss	s	string concatenation (force numbers to strings)

5.				
	round	nn	n	round to no. of decimal places
	<<	nn	n	left shift
	>>	nn	n	right shift
6.				
	in	ss	n	returns first position of substring, else 0
	notin	ss	n	returns 1 if no match for substring, else 0
7.				
	<	pp	n	less than
	>	pp	n	greater than
	<=	pp	n	less than or equal
	>=	pp	n	greater than or equal
8.				
	=	pp	n	equal to
	==	pp	n	equal to
	<>	pp	n	not equal to
	!=	pp	n	not equal to
9.				
	and	nn	n	logical and
	&&	nn	n	logical and
10.				
	or	nn	n	logical or
	||	nn	n	logical or

Items

Items are the values on which operators work.

The following are used to describe items in the item list:-

<name>	the name given to a variable
<num>	a numeric quantity in decimal format - e.g. 79
<hex num>	a numeric quantity in hexadecimal format - e.g. 4F
<binary num>	a numeric quantity in binary format - e.g. 01001111
<string>	any string of characters - may also include numbers

Items may be any of the following:-

<name>	variable name
<name>[<num>:<num>]	substring of variable
"<string>"	string constant
'<string>'	string constant
<num>	decimal number
&<hex num>	hex number
#<hex num>	hex number
%<binary num>	binary number

Coercion of types

Variables are automatically converted to the appropriate type required by the operator, where possible. In cases where this is not possible, an error message will be displayed at the bottom of the screen, stating where the invalid operation was found.

Note: In some cases an expression may be valid, but the items being evaluated may not be valid in all circumstances. e.g. 'total/items' is valid unless 'items' has a value of 0, in which case a 'division by zero' situation arises and an error results.

26. Mail Merging

What is Mail Merging?

In its simplest form, mail merging is the process of incorporating text from one file into the text of another file. One of the most common forms of mail merging is incorporating names and addresses from a data file into a standard letter, in order to create a "personalised" letter for each person. There are, however, many variations and the next few sections will describe these, starting with simple examples and gradually introducing the more advanced features of mail merging with Protext.

The mail merging facilities of Protext are very advanced and will permit the carrying out of virtually any sort of mail merge. Selective, conditional and alternative text merges are all possible, as are arithmetic calculations and use of the current date and time. Each will be covered in detail in the following sections.

Despite being a very advanced program, Protext has been designed in such a way that straightforward mail merging is very simple to understand and use, whilst at the same time facilities are incorporated to permit the most extensive of mail merges and calculations.

Because mail merging can be a complex subject, this part of the manual is written on the basis that the user has no knowledge of the subject and instead of listing all the facilities at once, introduces them gradually. In this way it is easy to carry out simple mail merges and with experience, to progress to more complex examples. At the back of the manual is a technical appendix which details all the stored commands in alphabetical order.

Throughout the chapter on mail merging, many examples are given, with the recommendation that they be typed in, tried and saved. On occasions earlier examples will be used again and expanded to demonstrate the use of new commands. Many of the examples can be used as 'template' files and modified at a later date to suit the needs of users.

One of the features most commonly found in connection with mail merging is the use of data files and the next section covers data files in detail.

Data Files

Data files are a very important aspect of mail merging, so before looking at mail merging, it is important to know exactly what they are and how they are created.

What is a data file?

A data file is a collection of items of information which may be anything from names, addresses and telephone numbers, to paragraphs of text, or collections of numbers.

Data files can be created in a number of ways. The simplest example would be one created by Protext, in which case the data would be entered using Protext in the normal way and then saved to a file on disc.

Most good databases and many spreadsheets are capable of creating data files suitable for use by Protext, but in a few cases, programs may create files which are largely, but not entirely, suitable. The REPLACE command may be used to easily convert such files into a suitable format. Details are given in the section 'Exporting data from other programs'.

A third source of data is 'output' from suitable Basic programs, written to process and store data.

Any ASCII file may be used by Protext as a data file, providing the data is laid out in a suitable form.

Data file format

A data file will usually contain a number of records, each of which will contain one or more fields of information.

For example:-

A data file might contain a list of names and addresses, in which case the name is considered as one field and each line of the address as a further field. All the fields that constitute one complete name and address are considered to be one record. All the records constitute one data file.

How Protext reads data

When Protext reads from a data file, it reads characters from the beginning of the file, until it finds a comma or a carriage return character, either of which it will take to indicate the end of the first field. It will then take the next character it finds to be the start of the next field and repeat the process until the appropriate number of fields has been read. Leading spaces in a field are ignored.

The only exception to this is when the first non-space character of a field is found to be either a single or a double quotation mark, in which case Protext will continue to read the characters until it finds matching quotation marks, which it will then take to indicate the end of that field. Any quotes of the alternate sort, or commas, contained within the matching quotes will not be treated as end of field markers in this case, but as part of the text. In addition, any leading or trailing spaces inside the quotes will be retained as part of the text. It can therefore be seen that on occasions it may be desirable to 'wrap' fields in quotation marks to ensure that fields do not get broken up by commas and quotation marks in the text.

Normally each record will be separated from the next by a blank line and this will be used by Protext to indicate the end of a record.

Occasions can arise where one or more of the records may contain fields which have no information, for example when one of the fields is for telephone numbers. Unless the absent field is the last field in the record, it is vital that the field is not left out of the record and empty fields can be represented either by an empty line, or alternatively, the dollar symbol '$' may be used to represent an empty field. When reading data, Protext understands a single dollar symbol to indicate an empty field and when it has been read, converts it to an empty string. One of the advantages of using the dollar symbol is that it makes it easier to view the data file and see where each record starts and ends, but there are other reasons why it may be preferable, and these will be covered in later sections.

Note: Whilst it is preferable to have a blank line between each record, some programs produce 'Comma delimited' export files. These files have all the fields of a record on one line, with each field wrapped in quotation marks and separated by a comma and sometimes without a blank line between each record. As long as all the records contain the same number of fields, even if some of them are empty, these data files are still quite usable without any conversion. In the event that they contain unequal numbers of fields, they may still be used after conversion with the CONVERT utility (See 'File conversion') to insert a blank line between each record.

Examples of valid data file records.

The following are all examples of valid formats for data files suitable for use with Protext and could all be used in the same data file, although it is preferable to use only one format in any data file, if only for reasons of ease of creation and alteration.

J Smith
$
The Cottage
Little Deighton
Nr York
West Yorkshire

D Grey

15 Ambleside Road
Moortown
Harrogate
Yorks

"M Black"
"0539 412345"
'"The Leys"'
"Cartmel Fell"
"Ulverston"
"Cumbria

"P Brown","","16 High Street","Sandylands","Kendal","Cumbria"

R White, 9923 46143, 24 North Ave, Hampton, Keswick, Cumbria

J Green, $, 64 Park Road, Stockport, Cheshire, $

Note: In the examples where the fields are wrapped in quotation marks, a 'null' field may be represented either by two consecutive quotation marks or a dollar symbol wrapped in quotes.

Exporting data from other programs.

Note: This section is included here as it concerns data files, but it is suggested that it is only studied in detail after the rest of this chapter has been read and understood.

The most common source of data, other than data files created with Protext, is in the form of output from a database program. There are numerous databases available most of which provide 'export' facilities. The degree of flexibility of output varies from program to program, as also does the terminology used in these programs.

The object of this section is not to give specific details on how to export data from each individual program, but to indicate the general principles involved and the terminology used. It will also be noted that some programs make it easier to export data than others and if a database is being purchased for use with Protext, consideration should be given to this feature as well as to the normal database features.

Many programs have an 'Export' option or an 'ASCII save' option and these are generally the options required to create a suitable file. Often this option will lead to a series of questions about what form the data export is to take. Sometimes the option will be provided to 'wrap in quotes' or 'Comma delimit' the fields.

'Comma delimited' is a term frequently used in database export facilities and if this is selected, all fields will be wrapped in quotation marks and each field will be separated from the next field by a comma. Each record will occupy one line.

'Wrapped in quotes' is a similar option, but instead of fields being separated by commas, each field will be wrapped in quotes on a separate line.

If there is a possibility that the contents of the database fields might contain any commas or quotation marks, then either of these options should be selected, preferably the 'wrapped in quotes' option.

Even if neither of these options are available, it is possible with some databases to create a 'Report form' which may be used for export and in this case it is usually possible to put quotation marks round the fields in the report, with each field on a separate line and then to export the data using that report form, so that the quotation marks are incorporated as well.

'Record separators' is another possible option. If selected then each record will be separated by a blank line in the data export file. If this option is not available, the same effect may usually be achieved by specifying a blank 'dummy' field in the database and then having it as the last option in the list of export fields. Alternatively, if the database uses the 'report form' type of export, then a blank line at the end of the form may be sufficient.

Some programs are only capable of sending 'raw' data to a data file. In these cases, it may be necessary to use the Protext 'REPLACE' command on the resultant data file, in order to wrap fields in quotes or insert blank lines between records. Wrapping fields in quotes is desirable in case there are any commas or quotation marks within the fields, which would otherwise be treated by Protext as indicating the limit of a field.

Due to the wide variations in facilities, it is not possible to give specific details for individual programs, but with the above information and a trial export of data it should be possible to make virtually all export files suitable for Protext. If all else fails, it will be possible to load the data file into Protext and modify it as necessary by use of the REPLACE option.

Wrapping in quotes

```
REPLACE  !.  '"!."'  GA
```

will insert double quotes at the end of every line and the start of the following line. It will only be necessary to add a double quote at the start of the first line and remove a set at the end of the file. This would be ideal for wrapping fields in a data file which had each field on a separate line. It may be a good idea to replace all double quotes with single quotes before doing the above 'quote wrapping' so that all quotes in the fields are of the alternate type.

Inserting blank lines

For a 'comma delimited' file with no blank lines between records:-

```
REPLACE !. !.!. GA
```

will insert a blank line between each record.

A special note:

Many people will have discovered the frustration of trying to transfer data from an existing database program to a new database, where the format of the export files is incompatible with the new database. Protext may be used to reconstruct the data file in the required format by creating a merging file to read the data from the existing file and then using the SAVEA command to send the output to a new file in the required form. A simple example of this process is given in the Appendix listing examples of mail merge files, together with an example of converting dates into a different format.

Using Prodata with Protext

There should be no problems using Protext and Prodata as both programs are very flexible about the form of output they can produce and accept. Usually the data would be exported from Prodata using the default export options (just press RETURN to each question). The Protext '>RU' command will then be used to read the data (see below). The 'wrap in quotes' option should be used if the data in Prodata contains commas.

Simple Mail Merging

At many stages throughout this section of the manual it will be suggested that text and data files are saved. There are two reasons for this. Firstly, because many of the examples will be used at a later stage and expanded as new commands are introduced and secondly, many of the examples will readily convert into template files, which can easily be modified to create mail merge files for a variety of purposes.

The concept of keeping template files has been mentioned in the chapter on stored commands and is equally valid, if not more so, with mail merging documents. For many people the example files created in this chapter will be the only templates they need, requiring only a few changes to suit different purposes. In this way, it is possible to quickly create new documents without the need to start from the beginning each time.

Getting started

The principle on which Protext carries out mail merging is very simple. At the start of a document, Protext opens a specified data file and reads the contents of each field, allocating them to variables. When all the specified variables have been filled, Protext continues printing the document, inserting the contents of each variable into the text, wherever a variable reference is found. By the time the end of a document is reached, all the variables for a field must have been read. The process is then repeated, with further fields being read from the data file and the next copy being printed.

Variables are inserted into the text, at the appropriate position, by placing an ampersand ('&') immediately before and immediately after the variable name. For example, to insert a variable called 'name' into the text, it is simply entered as '&name&'. There is no restriction on the number of times each variable may be used in the text.

If variables are positioned in the text using tab markers, then the contents of the variable will be justified about the tab location when printed, in the normal way.

Note: Users should note that the contents of any variable positioned in the text using the decimal tab will automatically be positioned accordingly in the course of printing, enabling automatic justification of any tables of figures contained in a data file. Ensure that there is enough room between the tabs for the variable contents.

One of the most common forms of mail merging is where a number of letters are to be sent to individuals and their name and address is required on each letter.

Creating a Data file

At this stage, it would be a good idea to create a small data file, with only five or six records, for use with the examples which follow. The easiest way to create a small data file is simply to type it in, using Protext. Enter the name on the first line and press RETURN then type in the address on separate lines, using RETURN at the end of each line. For the purposes of the example, make the records a mixture of three and four line addresses and for the time being, do not put any punctuation marks or quotes in the fields. After entering the last address line of each record, press RETURN a second time, so that there is a blank line between each record, and do the same at the end of the file. Save the file with the name 'DATFILE1'.

Reading the data

Initially, there are only three stored commands that may be required, two of which will be used in the majority of cases of mail merging. These are:-

>DF file {file}	Define data file.
>RU var {var}	Read variables unconditionally.
>RV var {var}	Read variables.

Note: The use of curly brackets indicates that optionally, one or more extra items may be specified. 'file' may be any valid filename and 'var' may be any word starting with a letter (including letters with accents) and may contain letters, numbers, '.', '_' and '?'.

DF	Define data file or files.
Examples:	>DF datfile >DF file1 file2 ... filen
Description:	The files should contain data which is going to be used for mail merging. Usually there will be only one file, but several are permissible. When each file is completed, the next one will be opened for reading. This command is used to tell Protext the name of the file to be used, and so must occur before reading data from the file.
Note:	If more than one DF command is required in a document, a CF command must first be used to close one file before opening the next (See Advanced mail merging).
Note:	The file name may be a full pathname.
RU	Read variables unconditionally from the data file.
Example:	>RU var1 var2 ... varn
RV	Read variables from the data file.
Example:	>RV var1 var2 ... varn

The number of variables named in these commands must be the same as the maximum number of fields in a data record, usually with one added for the blank line between records. They need not all be in one 'RU' or 'RV' command, but by the time a complete document has been printed once, all the fields for a record must have been read.

'RU' and 'RV' are very important commands, being the means of extracting data from a data file for use during mail merging. Whilst both commands are used in a similar fashion, with the same syntax, there are important differences between the way that the two commands work, the choice of which command to use being largely determined by the format of the data file. A thorough understanding of both these commands is important and the differences in their uses will be discussed at various stages in the manual.

Clear any text from the computer, using Protext's 'CLEAR' command and enter the following example program. Save this as 'EXAMPLE1', as it will be used in later examples and modified as further commands are introduced.

Example - simple mail merge to read names and addresses from a data file and insert them into a letter.

```
>CO    EXAMPLE1
>PL 24                                  ;for example purposes only
>CP OFF                                 ;single sheet
>DF datfile1                            ;define data file
>RV name addr1 addr2 addr3 addr4 dummy  ;rem read variables

&name&
&addr1&
&addr2&
&addr3&
&addr4&

Dear &name&,
    Thank  you  for  your  letter  about  the  insurance  for  &addr1&  &addr2&
&addr3& &addr4&. We think that you will find the rates quoted......
```

Before trying out the example, it is worth studying it to see what has been done. The first line is a comment line and is used as a reminder of what the document is. The 'PL 24' command in the second line and the 'CP OFF' in the next line are present only for the purposes of this example, in order that when it is printed to the screen, each record will be printed and then stop, so that it can be studied.

'DF datfile1' defines the name of the data file, so that Protext can open it for reading. The following line, which in this first example makes use of the 'RV' command, is there to read the various fields of the record and allocate them to the variables. It is good practice to choose variable names that are related to the data that the variables will contain. Special notice should be taken of the variable 'dummy', which will be used in the majority of mail merges and is used to read the empty line between each record.

'RV' has been used in this example for a reason. One of the special characteristics of 'RV' is that when it reads from the data file, it stops reading any more data as soon as it finds an empty field and automatically gives a null value to any remaining variables on that 'RV' command line which have not yet been allocated values.

With the data file that has been created, this is important, as some of the records have less fields of data than others. If the record was one that had only three address lines and all the variables were to be read and allocated, then the variable called 'addr4' would contain the empty line between records and 'dummy' would contain the name field from the next record, so that when the next record was read, the first field would be allocated to the variable 'name', which in this case would be the first address line. When using 'RV', as soon as the blank line is read, no further fields are read and any remaining variables are given a null value.

The remainder of the example is the text that will be printed and, as described earlier, wherever the data is required, the appropriate variable is entered in the text, wrapped in its markers.

To try out the example and save wasting paper, use the PRINTS (PS) command to direct the output to the screen instead of the printer. PS is particularly useful with mail merging, as it allows a check that everything is happening as expected, before printing it out properly.

One thing that will be noticed is that even though some of the addresses may have only three lines, Protext closes up the blank line and does not leave unsightly gaps. The space between the last line of the address and 'Dear..' will always remain as one line.

If, for any reason, it is desired to retain the gap, Protext has an alternative set of variable markers, '!' to replace the '&' and these can be used instead. Example: '!addr4!'.

Even more noticeable is the way that the remainder of the paragraph of text containing the address is automatically reformatted during printing, ensuring continuity of the text.

The two commands used so far are the only ones required for the simplest of mail merges.

Comparing RV and RU

In order to see the different result obtained by using the 'RU' command, change the 'RV' command to 'RU' in 'EXAMPLE1' and repeat the PRINTS. When using 'RU', the names and addresses will soon get confused and mixed up.

Load the data file back into Protext and go through the records, inserting a blank line at the end of any record that is missing the fourth address line, making each record effectively the same length. Save this amended data file with the name 'DATFILE2'.

Reload the text file 'EXAMPLE1' and change the name of the data file specified by 'DF' to 'datfile2'. Carry out another print, using 'RV' to read the data and everything will get confused again, but when replaced with 'RU', everything will work correctly.

From the above, it should be seen that the main use of 'RU' is with data files which contain records with the same number of fields in each record. Many database programs construct data files of this sort, which are suitable for use with Protext. Except for the simplest of data files, containing only names and addresses, it is normally preferable to make sure that all records are of the same length, even if it means that either blank lines or lines containing the '$' marker, are used to signify an empty field. Some programs permit you to specify default text, to be used in the event of an empty field being found, in which case '$' would normally be specified.

'RV' can also be used to read data files which contain a constant number of fields, but if any of the fields in a record are null, then 'RV' will interpret these as the end of a record and give a null value to the remainder of the variables, which is not the required result. To see this in action, reload 'DATFILE2' and insert some telephone numbers after the name and before the address. Leave one or two of them as blank lines and save as 'DATFILE3'.

Reload 'EXAMPLE1' and alter the data file name to 'datfile3'. Change the 'RV' line to 'RU' and insert a variable called 'tel'. The line should now look like:-

```
>RU name tel addr1 addr2 addr3 addr4 dummy  ;rem read variables
```

Save this file with the name 'EXAMPLE2' and carry out another PRINTS. Everything will work correctly, as 'RU' reads all the fields, including any empty ones.

Note: When amending earlier examples, to create new ones, do not forget to alter the title in the first comment line to suit the new example number.

Note: It is necessary to read the telephone field and allocate it to a variable, even though it is not actually required for the document. It is essential to read all the fields of a data record, even though only certain of them may be required, otherwise Protext will get out of step with the data records.

Now change the 'RU' to 'RV' and repeat the print. This time things will get out of order again, as 'RV' will cause Protext to stop reading fields as soon as it comes across an empty field.

There are two ways to get round this problem with 'RV', the simplest one being to use the '$' marker, as described in the section on Data files, in place of an empty line wherever there is an empty field in a record, but this is not always possible with data files created by other programs. The second method requires the splitting of the 'RV' command on to a number of lines and is covered in detail in the section on advanced mail merging.

As a general rule, 'RU' is used to read the contents of data files having records which contain a constant number of fields, whatever the format of the file, whilst 'RV' is used for simple data files where the last field in a record may, or may not, be present and for a number of specialised uses, discussed later in the manual.

Creating data by other means

So far, the only way that it has been possible to insert text into a document has been by reading it from a data file, but there are two commands which enable data to be provided in other ways.

```
>AV (message) var (number) {(message) var (number)}     Ask variable
>SV var = string of text {var = string of text}         Set variable
```

Note: The use of normal '( )' brackets indicates that the parameter is optional, whilst curly brackets, as before, indicate that one or more extra items may be specified. 'message' is a piece of text to be displayed on the screen and 'number' is a number between 0 and 255.

AV Ask for variable from keyboard.

Examples:

```
>AV "Name",name,20,"Address line 1",addr1
>AV name,addr1,addr2,addr3,addr4
>AV "Continue (Y/n)" key 0
```

Description: The messages are displayed on the screen as prompts. If no message is specified for any particular variable then the variable name is displayed on the screen followed by a question mark. Text may then be typed in, and this is assigned to the variable. If 'number' is specified this sets the maximum length of text that may be entered. The maximum length permitted is 255 characters, and this is the default value. If the number is zero a single key press will be read and assigned to the variable - if RETURN is pressed the empty string is assigned.

SV Set variable to a value.

Examples:

```
>SV firstname="Peter"
>SV name=firstname+" Smith"
>SV counter=1
```

Description: This command automatically sets a variable to a given value, which can then be used in the text in the same way as other variables. The 'string of text' may be either a string of characters, which must be enclosed in quotes, or it may be another variable, in which case the variable name is used, or it may be a combination of both, in which case they are joined with a '+' sign, as in the second example. Valid decimal numbers may also be used, either with or without quotes.

There are many other things that can be done with the 'SV' command, but they are beyond the scope of this introductory section and are discussed in detail in the later section on variables.

The easiest way to see the effect of these two commands is to modify 'EXAMPLE2', used earlier. The following two lines should be inserted after the line which starts '>RU'.

```
>AV "Enter Date " date
>SV year="1986"
```

and the following line after 'Dear &name&'

```
>CE  Renewal date - &date& &year&
```

Save the modified file as 'EXAMPLE3' and use PRINTS again, to see the results.

The main commands concerned with simple mail merging have now been covered but there is one final command which should be mentioned here which, whilst not strictly being mail merging in the normal sense, is a simple and yet very powerful command.

Inserting separate text files

'IN' is a very useful command which permits the merging of commonly used pieces of text into a document. For example, the user's address could be kept in a file on disc called 'address' and instead of loading it into memory each time a letter is written, it would only be necessary to put '>IN address' at the start of each letter. As another example, if the same paragraphs of text are frequently used in business letters, these could be saved on disc with appropriate names and called as necessary.

IN	Insert file into text.
Example:	>IN filename
Description:	This command may be used at any point in the text of a document and the named file is read from disc and the contents printed. When the end of the file is reached, printing of the original file continues at the line after the IN command. The file is not loaded into memory, merely printed. Any number of 'IN' commands may be issued.
Note:	Drives and paths may be specified preceding the filename, in the normal fashion.

Another use of 'IN' is to permit the continuous printing of a number of files as one document. A text file consisting of the following entries will, when printed, result in 'file1' being printed, followed by 'file2' and finally 'file3'.

```
>IN file1
>IN file2
>IN file3
```

One of the advantages of this is that page numbering, headers, footers, or margins set at the start of the document will follow through correctly, unless altered by any stored commands in the files being printed. In this way, it would be possible to print out a whole book, from beginning to end, in one run, with headings and page numbers running correctly through.

Further uses of the 'IN' command are detailed in the section on 'Advanced mail merging'.

Address labels

Finally, before moving on to more advanced mail merging, this very common and straightforward use of Protext should be mentioned.

```
>CO Address labels
>PL 9           ; Number of lines to repeat label (6 per inch)
>FF OFF         ; Don't send form feeds after each label
>ZM             ; Set all margins to zero
>DF datfile1    ; data file to read
>CO Next six lines read contents of one or two records
>RV name,addr1,addr2,addr3,addr4,dummy
>IE
>SV nexname="",  nexaddr1="", nexaddr2="", nexaddr3="", nexaddr4=""
>EL
>RV nexname,nexaddr1,nexaddr2,nexaddr3,nexaddr4,nexdummy
>EI
>---------------------------------------!----------------------------------R
!name!                                  !nexname!
!addr1!                                 !nexaddr1!
!addr2!                                 !nexaddr2!
!addr3!                                 !nexaddr3!
!addr4!                                 !nexaddr4!
```

There are several points that should be noted with the above example.

a). 'PL' should be set to a value equal to the number of lines that are required to move from the top of the first label to the top of the next label. The normal setting of a printer is six rows to an inch. In the above example, the setting is therefore for address labels with a one and a half inch repeat.
b). 'ZM' must be used to set all margins to zero.
c). 'RV' is used, as 'datfile1' contains variable length addresses.
d). Full explanation of IE, EL and EI is given in the following two sections.
e). Note also that there should be no blank lines after the stored command, which is used to force a new page and the labels to be advanced to the start of the next label.
f). The alternate set of variable markers, '!', have been used to ensure that even if a field is empty, the empty line will be printed, ensuring that the printer always moves on the correct amount to the next row of labels.
g). TAB should be used to position the right-hand column of variables so that the addresses are lined up correctly.

Conditional Printing and Mail Merging

Conditional printing and conditional mail merging are really two different things, but, as they are frequently used together, both will be covered at the same time.

A conditional mail merge is one where text will be either printed or not printed, depending on whether certain conditions are met. It may affect whether all or only part of the text is printed, or even whether one piece or an alternative piece will be printed.

A little thought will give some idea of the scope of these commands and once the principles are understood, some time spent experimenting will soon bring to light numerous possible uses. There are certainly far too many uses to cover all of them in this manual.

There are four commands which are directly concerned with straightforward conditional printing, some of which can be used on their own, but most of which are used in conjunction with each other.

>IF condition	If condition is true then print block.
>EL	Else print other block if IF condition false.
>EI	End of an IF block.
>SK condition	Skip copy if condition true.

Before considering these commands, it is important to have a thorough understanding of what is meant by 'condition'. A condition is where a string (piece of text), a number, or a variable, is compared with another string, number, or variable. The result will be either 'true' or 'false'. For the purpose of these commands, there are eight possible comparisons that can be made:-

=	equal to
< >	not equal to
<	less than
< =	less than or equal to
>	greater than
> =	greater than or equal to
IN	is contained in (e.g. "jo" IN "Mr Jones" is true)
NOTIN	is not contained in

The IN operator returns the position that the first string is found within the second string, or 0 if not found. For example "town" in "newtown" will return 4.

Decimal numbers are compared numerically. A numeric argument is one which the program recognises as a valid decimal number, such as '37' or '5.4'. If a valid number is found, it will be compared numerically, otherwise it will be compared character for character. 'IN' and 'NOTIN' will automatically carry out a character comparison, but the other comparators will attempt a numeric comparison first, before defaulting to a character comparison if the comparison is numerically invalid. For the purposes of a character comparison, no account is taken of whether a letter is upper or lower case and both are considered equal.

The first and simplest of these commands is:-

IF	Print if condition true.
Example:	>IF "Married" NOTIN status >IF firstname IN fullname >IF age>40 ; (i.e. true if age is over 40)
Description:	The condition is tested and if true the block of text which follows is printed (and any stored commands acted upon). The block of text extends until either an 'EL' command, an 'EI' command, or the end of the text. If the condition is false then everything within the block is ignored, including stored commands.
Note:	IF blocks may be nested to a depth of 7.

EI	Endif. End of IF block.
Description:	This command marks the end of an 'IF' block. If the 'IF' block was nested within another, then printing reverts to the remainder of the previous 'IF' block, otherwise unconditional printing resumes.
Note:	Every 'IF' command must have a matching 'EI' command.

Load 'EXAMPLE3' and add the following lines to the end of the text, save as 'EXAMPLE4' and again use PRINTS to see the effect:-

```
>IF tel>""
        If you would be kind enough to telephone us....
>EI
```

A check is made to see whether the contents of 'tel' are greater than nothing and any of the records which contain a telephone number will have the extra line of text added to the document, but if the telephone field is empty, then it will be omitted.

EL	Else.
Description:	The block of text following this command is printed if the condition in the previous IF command was false, but is ignored if that condition was true. The block extends until an EI command or the end of the text.
Note:	IF blocks may be nested within EL blocks.

'EL' is a very useful command, as it permits the inclusion of an alternative piece of text, to be used in the event that an 'IF' condition is found to be false. i.e. "IF this condition is true then print this block, ELSE print the next block".

Add the following lines to 'EXAMPLE4', immediately before the '>EI' command, save as 'EXAMPLE5' and again use PRINTS.

```
>EL
        If you would be kind enough to write to us ....
```

This time either one or the other message is printed, depending on whether a telephone number is present in the field or not.

'Nesting' has been mentioned in the previous commands and this is the process where one IF block is situated inside another. IF blocks can be nested to a depth of seven. It is not possible to have IF blocks overlapping each other and if this is attempted, either the results obtained will not be what was expected, or else an error message will be given by Protext, depending on the precise situation. The example below will give some indication of nesting blocks.

For example:

```
>IF tel>""                    ; (IF 1) if condition 1
>IF name="Smith"                  ; (IF 2) then if condition 2
    Your name is Smith and you have a telephone...

>EL                               ; (EL 2) else if not condition 2
>IF name="Jones"                      ; (IF 3) then if condition 3
    Your name is Jones and you have a telephone...

>EL                                   ; (EL 3) else if not condition 3
    Your name is not Smith or Jones, but you have a telephone...

>EI                                   ; (EI 3) end condition 3
>EI                               ; (EI 2) end condition 2
>EL                           ; (EL 1) else if not condition 1
    You do not have a telephone...

>EI                           ; (EI 1) end condition 1
```

From the above example, it can be seen that condition 2 is nested completely inside condition 1 (between the IF and the EI commands). In turn, condition 3 is nested entirely within the EL part of condition 2.

It may be found convenient when creating documents of this sort to do so in a particular order and to add comments to each command line. In the above example, each level of nesting has been marked with the command with which it is associated and the comments have been tabulated across the document to make it clear which commands are related. It may also be found convenient to insert all of the related IF, EL and EI commands at the same time and then to insert the text within them as appropriate.

SK Skip printing if condition true.

Examples:
```
>SK "Mr" IN name       ; If name contains 'Mr', not printed
>SK add4="London"      ; If add4 is "London", not printed.
```

Description: The condition is checked and if true the remainder of the text is skipped. This allows the selective use of data from a file.

Note: The same effect could be achieved by the use of 'IF', but 'SK' is preferable where it is required to omit printing the remainder of the text for that record, as the effect is immediate, whereas 'IF' will continue to check through the document for further commands, until the end of the document is reached.

The easiest way to see how this command works is to insert the following line into 'EXAMPLE5' immediately after the '>RU' line and save it as 'EXAMPLE6', before printing.

```
>SK addr4 > ""
```

This line tests to see whether the variable 'addr4' contains anything or not and if it does, then the condition is true and the remainder of the document will be skipped. In this case, as the test is made before anything is printed, the complete record will be skipped. The effect of this is that only records with three address lines will be printed. Another example would have been to use '>SK tel=""', which would have resulted in only records containing a telephone number being printed.

The commands covering the simplest forms of conditional mail merging have now been covered, but there are several other commands which extend the possibilities further and these are covered in the section on advanced mail merging.

To make the most of Protext it is recommended that some time is spent experimenting with the various commands covered so far, before continuing with the section on advanced mail merging.

Advanced Mail Merging

Many new commands were described in the earlier sections and here further new commands will be introduced, as well as extensions to a number of commands already covered in their simpler forms.

More new commands

>CF	Close data file.
>IE	If data file exhausted then print block.
>ID var	If variable defined then print block.
>IU var	If variable undefined then print block.
>RP	Repeat.
>UN condition	Until the condition is true.
>WC	Write file close.
>WF filename (A)	Open file for writing.
>WF ON/OFF	Turn writing to file on and off.
>WM text	Write message to file.

Closing data files

In earlier sections the 'DF' command was used to open data files for reading. Normally a data file will be closed automatically when the last of the data has been read, but there are a number of occasions when it may be preferable to close a data file prematurely.

One of the restrictions mentioned previously was the fact that it is only possible to have one data file open for reading at a time. For many purposes this will be no problem, but it can be useful to be able to open another data file, having extracted the required information from an earlier file. 'CF' is provided expressly for the purpose of closing one data file, in order that another data file can be opened in a subsequent command.

CF	Close data file.
Example:	>CF
Description:	CF closes a previously opened data file, permitting a further file to be opened for reading.
Note:	A CF command must be issued between two DF commands, otherwise the second DF command will be ignored.
Note:	A data file closed by the use of CF can be reopened at a later stage with a further DF command, but the contents will be read from the start of the data file each time.

Special conditional commands

The use of the 'IF' stored command has already been explained in the section on conditional printing and mail merging, but in addition to 'IF' there are three further commands which can be used instead of 'IF' for specialised purposes. These are:

IE	If exhausted.
Description:	The data file is checked and if it contains no further data, then the block marked by 'IE' is printed. It is usually positioned at the end of a document so that when the current print is completed, the data file will be checked to see if there is any further data. This command is particularly useful for displaying a summary when a mail merge run is completed.
ID	If defined.
Example:	>ID var1
Description:	ID is a special form of the IF command, which returns 'true' if the variable has been defined, and 'false' if not. If true, the text following will be printed, or any stored commands acted upon, until the matching 'EI' is found. This command is very useful when used in conjunction with AV.
IU	If undefined.
Example:	>IU variable
Description:	IU is the opposite of ID, in that if the variable has NOT been defined, then the condition is true.

A variable has been defined if it has already been given a value earlier in the text printing, by any of the commands AV, RU, RV, or SV. This allows, for example, an AV command to be executed only once and the resultant value to be used for the remainder of the mail merging.

The following example gives a demonstration of the use of both the IE and IU commands. Reload 'EXAMPLE4' and make the following alterations:-

Insert this line before '>AV "Enter date" date':

```
'>IU date'
```

Insert these lines following '>SV year="1986"':

```
'>SV total=0'
'>EI'
'>SV total=total+1'
```

Insert these lines at the end of the file:

```
'>IE'
'>SV total=total[w1]'
'>DM You have printed &total& copies.'
'>EI'
```

Save the text as 'EXAMPLE7' and using PS it will now be found that at the start of printing, the date will be requested. This has the effect of defining the variable 'date', so that in subsequent copies, as 'date' has now been defined, the date will not be requested and the existing value will be used for all further copies. Finally, when all copies have been printed, a message will be displayed on the screen stating how many copies were printed.

'ID' is similar, but the text which follows it will only be printed if the variable has already been defined.

Repetitive looping

Protext also provides the means to carry out repetitive actions in a document. These are carried out using the following commands.

RP	Repeat.
Example:	>RP
Description:	'RP' marks the start of a block of text that will be repeated until a condition specified in 'UN' is found to be true. Normal printing will then continue.

UN	Until a condition is true.
Example:	>UN "N" < name
Description:	'UN' marks the end of a Repeat - Until loop. When Protext finds this command, it checks to see if the specified condition is true and if not, jumps back to the earlier 'RP' command. This sequence will repeat until the condition is true, at which point the loop will be exited and normal printing of the rest of the document will continue.
Note:	A Repeat - Until loop will always be carried out at least once.
Note:	When RP UN is used to carry out printing, it should be remembered that any natural page breaks may be altered and it may often be desirable to use a forced page break ('>PA n') at some stage after completion of the loop.

These commands can be used in a number of ways, either to repeat a block a set number of times, or until a certain condition is met.

Example to create a data file:

```
>ZM
>RP
>AV "Enter name " name
>IF name>""
>AV addr1 addr2 addr3 addr4
>SV dummy=""
!name!
!addr1!
!addr2!
!addr3!
!addr4!
!dummy!
>EI
>UN name=""
>ST
```

In the above example, the name is requested and checked to see that it contains text. If so, then the address is requested and the text printed, before looping back to repeat the process. If 'name' contains nothing, then the check in 'UN' will be found true, the loop will be left and printing will stop. Using SAVEA, to print to a disc file, will create a file suitable for use as a data file.

Note: The alternate markers, '!', have been used to ensure that the same number of lines are sent to the data file each time, even if a field is found to be empty.

Example to request information a given number of times.

```
>---------------------!----------!-----------!--------!----------R
Description of goods  Unit cost  Quantity    Discount
>SV counter=0
>RP
>AV descrip cost number disc
>SV counter = counter+1
>--------------------------.----------.-----------.--------------R
&descrip&            &cost&   &number&      &disc&
>UN counter=10
```

In this example a variable called 'counter' is set to a value of zero and when the text is printed, requests are made for a description of the goods, the quantity and the discount. The value of the variable is increased by one and the loop repeated until the value of 'counter' reaches ten, when the loop will be left and the remainder of the text printed.

Writing to a disc file during printing

Protext provides the facility to write part or all of a document to a disc file during the course of printing. There are three commands which provide this facility.

WF Open write file to disc.

Examples:

```
>WF filename
>WF filename A
```

Description: A file of the specified name will be opened for writing. The '>WF ON/OFF' commands determine whether the text is written to the file or not, when printing takes place.
If the 'A' suffix is used, any existing file of that name will have the data appended to it.

WF ON/OFF Turn on and off writing to the file.

Examples:

```
>WF ON
>WF OFF
```

Description: When printing takes place, if writing is turned on, all text is sent to the file, including stored commands, printer control codes. rulers etc. at the same time as the document is printed normally. 'WF' may be turned on and off as many times as required during the course of a document.

WC Write file close

Description: Closes the file being written to by the above commands. If no WC command is given the file will be closed when printing finishes.

This is an extremely powerful function as it can be used to create individual documents, or parts of documents, during the course of a mail merge printout. When it is used, the file that is created is a Protext document and any stored commands, rulers or printer control codes will be sent to the file as well as the text that is being printed. The final result is a document which can be printed out as a normal file. The only difference from the original document is that where variables had been used in the original document, the new file will contain the data instead.

This command is particularly useful when used in conjunction with stored commands which request input from the keyboard ('>AV') and can be used to provide a copy of what was typed in at the keyboard, together with the remainder of the document. One specific use would be where Protext was being used to create invoices from information input at the keyboard. If the invoice is printed with 'WF' turned on, then a copy of the invoice will be provided on disc.

Care should be taken when using this command to ensure that the appropriate ON and OFF commands are positioned in the correct places in the document. If the whole document is printed with 'WF' turned on, then the result will be a file containing all the stored commands, except the 'WF' commands, as well. If the WF commands are positioned only around the text and are turned off when stored commands are being printed, then the result will be a facsimile of what has been sent to the printer, including printer control codes. It can be seen that it is possible to use these commands in a number of ways.

The following examples will give some indication of how the commands are used:-

```
>WF "agename.lst"
>SV counter=0
>RP
>SV counter= counter+1
>AV "Enter name" name "Enter age" age
      This is a list of ten people and their ages
>WF ON
      This person is called &name& and is &age& years old.
>WF OFF
>UN counter=10
      Remainder of text .......
```

This example will result in a file called 'agename.lst' being saved to disc, which will contain nothing more than consecutive lines stating 'This person is called nnnnnn and is yy years old', where 'nnnnnn' is each person's name and 'yy' is their age. The heading and the remainder of the document will not be sent to the disc file. The document which is printed will contain the heading, the name and age lines for ten people and the remainder of the text.

Compare this with the following example:-

```
>WF "copy.let"
>WF ON
>TM 5
>SM 8
>------!----------------------------------------------------------R
Dear Sir,
      Thank you for your letter.....
>WF OFF
```

This example, when printed, will create a file called 'copy.let', which will contain not only the text, but all the stored commands and rulers which are within the ON and OFF statements.

Displaying messages during printing

So far, there has been no means of knowing which record is next to be printed, until it is actually being printed. There are some occasions when it would be convenient to decide at printing time whether a record was to be printed, or even what was to be printed. Protext provides a number of commands to display messages on the screen. These commands are 'CS', 'DM', 'ST' and 'WT', which have already been described in the chapter on stored commands. What was not mentioned was the fact that variables could be used in the messages, thus providing a method of displaying their contents.

Examples:	>DM	The next record is &name&.
	>CS	Do you want to print the record for &name&?
	>WT	Print &name&? ESC to stop or any key to print.
	>ST	There are no more records to print called &name&.
	>WM	This letter is to &name&

In all cases the variable must have been defined before the message line and following the message line or lines, suitable stored commands should be inserted to take the appropriate actions.

Example: Modify 'EXAMPLE2' to the form shown below and save as 'EXAMPLE8

```
>CO EXAMPLE8
>PL 24
>DF datfile3
>RU name tel addr1 addr2 addr3 addr4 dummy
>CS    Do you want to print the record for &name&?
>AV " "print
>SK print<>"Y"
&name&
etc

>DM  Copy to &name& has now been printed
>WT  Press any key to continue
```

The above example goes through the complete data file, prompting with the name from each record and asking whether it is to be printed or not and displays messages after printing, to inform who it was for and to request the user to 'Press any key to continue'.

More on the RV stored command

In the earlier section on simple mail merging, many of the differences between the way that 'RU' and 'RV' function were described, as well as the fact that 'RV' was particularly suitable for use with the simplest of data files. Mention was also made of the fact that it could be used to read more complicated data files and in fact had definite advantages for certain purposes.

It will be recalled that one of the special features of 'RV' was the fact that as soon as it came across an empty field it ceased to read any further data fields and allocated a null value to any remaining variables on that command line. This was also one of the things that caused problems when trying to read records which contained empty fields, as it automatically gave a null value to the remaining variables, even though they were present in the record. This also caused the reading of the records to get out of step.

One solution was to make use of the special Protext indicator '$', to indicate to Protext that the field was present, but with a null value. In this way 'RV' would continue to read the remainder of the variables.

There is another solution to the problem, which can be used where the records contain empty fields, without the '$' marker, of the sort that may be created by some programs. The answer is to split the reading of the variables into a number of lines, in such a way that any field which may be empty is the last field on that command line. Several lines can be used if there is a possibility that a number of fields might be empty.

In the following example it is assumed that the data file contains records in which the second, sixth and seventh fields might be empty in some records and that each record consists of eight fields plus a blank line.

```
>DF datafile
>RV name tel
>RV addr1 addr2 addr3 addr4
>RV occupation
>RV salary dummy
```

When RV reads 'tel', if it finds it to be empty it will give it a null value, but as there are no more variables on that command line, Protext will move on to the next line and continue to read the next few variables. As soon as a null value is discovered, the remainder of the line will again be given null values and so on to the end.

More on the IN stored command

In the section on simple mail merging a description was given of the use of the 'IN' command to permit the consecutive printing of a number of files, whilst still maintaining continuity of page numbers, headers, footers etc. Mention was also made of the fact that 'IN' could be used to cause other files to be printed at any stage in the printing of a document, permitting the merging of commonly used blocks of text for example.

A third use, is the ability to use 'IN' at the end of a document to call another document. This can be repeated as many times as required. In general, a more satisfactory way of doing this, however, would be to use the previously mentioned list of '>IN filename' commands.

One further use of 'IN' which can be very useful is to use it in a recursive fashion, to call itself. This will permit the same document to be repeated as many times as required. The method of doing this is very straightforward and simply involves using the '>IN filename' command at the end of the file and then saving the file with the name used in the 'IN' command. In this way, when the end of the file is reached, it automatically calls itself. An example of this is shown in the appendix containing sample mail merge documents.

Making the most of Variables

There are a number of things that can be done with variables which have not yet been covered and this section introduces methods of manipulating the contents of variables in various ways, as well as introducing a number of other uses for variables.

Joining variables and text

When 'SV' was introduced, mention was briefly made of the fact that strings of text or variables could be joined together to make one variable. This can be particularly useful on many occasions. In 'EXAMPLE3', 'SV' was used simply to give a variable a value and then two variables were positioned next to each other in the text, so that when printed, they appeared as one. For example &date& &year&. It could have been done more simply by joining the two at an earlier stage and then using just the one variable in the text.

For example:

```
>SV year=" 1986"
>AV "Enter Renewal Date " date
>SV date=date+year
```

or more simply:

```
>AV "Enter Renewal Date " date
>SV date=date+" 1986"
```

Note: If variables which are to be joined together contain valid numeric values, then the result will be the numeric total of the two variables, which may not be the required result.

For example:

```
>SV num1="24", num2="10"
>SV num3=num1+num2
```

will result in 'num3' containing "34.00", not '2410'. If the result is intended to take the form of the characters being joined together, this can easily be achieved by using '$+' instead of '+':

```
>SV num1="24", num2="10"
>SV num3 = num1 $+ num2
```

which causes both 'num1' and 'num2' to be converted to strings first. See the following two sections for details of splitting variables and numeric calculations.

There are numerous uses to which this ability to join variables and strings of text can be put, but one in particular is worth mentioning.

No punctuation was used in the addresses of the earlier examples, with the exception of a full stop at the end of the first sentence, for a very good reason. Frequently the effect of 'personalised' circulars is spoilt by unsightly gaps in the text and misplaced punctuation marks. Protext takes care of the gaps automatically, but selecting the correct punctuation is a bit more difficult.

Commas and full stops could have been placed in the text, after each of the variables, so that they were printed. For many simple mail merges this would have worked satisfactorily as long as all the variables contained text. If some of the variables contained nothing, the result would be punctuation marks all on their own, or perhaps, two together. When printing 'EXAMPLE2', it may have been noticed that sometimes the full stop at the end of the first sentence was next to the address and sometimes it was separated because the last address line was missing.

Another way would be to put the punctuation in the data file as part of the text for each field, with the fields wrapped in quotes. In many cases this would work satisfactorily, but it has a number of disadvantages, not only because it becomes necessary to enclose fields in quotes in the data file, but because the situation can arise where the punctuation mark is not the one required in the text.

The solution is simple, once the principle is understood. Using 'EXAMPLE2' from the earlier section, if the record was one where the address consisted of only three lines, then the third (and last) line of the address would require a full stop, but if the address consisted of four lines, then the third address line would require a comma and the fourth a full stop. The easiest way to understand this is to look at an example of what is required. Load 'EXAMPLE2' and modify it before saving the modified version as 'EXAMPLE9':-

```
>CO    EXAMPLE9
>PL 24
>CP OFF
>DF datfile3
>RU name tel addr1 addr2 addr3 addr4 dummy
>IF addr4>""
>SV addr3=addr3+"," addr4=addr4+"."
>EL
>SV addr3=addr3+"."
>EI

&name&,
&addr1&,
&addr2&,
&addr3&
&addr4&

Dear &name&,
        Thank you for your letter about the insurance for &addr1&, &addr2&,
&addr3& &addr4& We think that you will find the rates quoted .......
```

In this example, it is known that 'name', 'addr1' and 'addr2' will always contain something, so it is sufficient to simply insert the punctuation in the text for these address variables. The 'IF' command checks to see whether 'addr4' is an empty variable. If not, it adds a comma to 'addr3' and a full stop to 'addr4', otherwise 'EL' puts a full stop on the end of 'addr3' instead. In this way, when printing takes place, the appropriate punctuation is printed as part of the variable. If there had been a possibility that an address could have consisted of only two lines, then an 'IF' command would also have checked to see whether 'addr3' had been a null string.

Splitting variables

As well as joining variables and text together, Protext provides the facility to split the contents of variables down into smaller parts, either by character or by word. This is achieved by specifying which characters or words are required in the following way:-

var[a:b]	from character 'a' to character 'b' inclusive.
var[a:]	from character 'a' to the end.
var[a]	character 'a' only.
var[Wa:b]	from word 'a' to word 'b' inclusive.
var[Wa:]	from word 'a' to the end.
var[Wa]	word 'a' only.
var[w-1]	the last word only.

If only numbers are specified, then Protext will split the contents on a character basis, but if the first number is prefixed with a letter 'W', then the variable will be split by words. The last option is a special case and will select the last word only.

Note: In this context, 'word' means any group of letters or numbers delimited by any of the following - a space, full stop or comma. Thus for a variable containing '12.45', var[w1] will give '12' and var[w-1] will give the result '45'

Examples:

If a variable called 'name' contains 'Robert Stephen Smith', then the following will be true:-

name[3:6]	is 'bert'
name[10:]	is 'ephen Smith'
name[3]	is 'b'
name[W1:2]	is 'Robert Stephen'
name[W2:]	is 'Stephen Smith'
name[W2]	is 'Stephen'
name[W-1]	is 'Smith'

Using 'Robert Stephen Smith' as an example again, the following values for the new variable will result:-

>SV newname=name[W1]+" "+name[W-1]	gives 'Robert Smith'
>SV newname=name[W2:]	gives 'Stephen Smith'
>SV newname=name[1:3]+" "+name[W-1]	gives 'Rob Smith'
>SV forenames=name[W1:2]	gives 'Robert Stephen'
>SV newname="Mr "+name[W-1]	gives 'Mr Smith'

It is also possible to use this form of splitting when checking conditions, using 'IF', 'SK' and 'UN'.

Examples:

>IF "Smith" IN name[W-1]	block is printed.
>SK name[W-1]>"M"	remainder is not printed.
>UN name[W1]="Robert"	repeat loop ends.

Using variables in stored commands

The use of variables in the stored commands concerned with displaying messages has been covered in the previous section, but it is also possible to use them in the majority of other stored commands, but in a much more sophisticated fashion.

The contents of a variable can be used in a stored command to define conditions, values and even other variable names in other stored commands. The easiest way to describe the use of this facility is by listing a few of the possibilities, but a full description of all the options is beyond the scope of this manual and it is suggested that some time spent experimenting with stored commands and variables will bring to light numerous possible uses.

Examples:

```
>AV "Name of file " filename
>IN &filename&

>AV "Start at page " startat "End at page " endat
>SA &startat&
>EA &endat&
```

```
>AV "No. of copies " copyno
>NC &copyno&

>AV "Enter Data file number" number ; if '3' was entered
>DF "datfile"&number& ;will open file called 'datfile3'
```

Numeric arithmetic

Numeric arithmetic is also possible with Protext and calculations can be carried out using the '+', '-', '*' and '/' operators. Compound calculations are permitted and the calculations are carried out using operator priority. Brackets may be used to force the order of calculations. Invalid numeric contents of variables will result in an error message.

Calculations can be carried out on the contents of variables, as long as they are of numeric type and the results may be stored in further variables. Variables that have been assigned a string value will, if possible, be converted to a number if accessed in a numeric expression. If any variable used in an expression is not numeric, it is assumed to be a string expression and the use of '-', '*', or '/' will give an error message.

Note: Some countries use the 'comma' instead of the 'point' as a decimal point. CONFIG has an option to configure Protext to accept the comma instead of the point. If this option has been set, then in all calculations involving decimal points, the comma should be used. To ensure that merge programs will run on any version use the 'DC' command to set the character.

DC	Set the decimal character.
Example:	>DC , >DC .

The ROUND operator

ROUND may be used to round the result of any calculation to a given number of decimal places. For example to calculate the VAT on an amount and round to an exact number of pence:

```
>SV vat = (amount * 0.15) round 2
```

ROUND should be used when comparing two numbers for equality any time that fractions or multiplication or division are involved in the calculation.

DP Set number of decimal places to be displayed

Examples:
```
>DP 0
>DP 5
```

Description: By default all numeric variables are displayed to 2 decimal places. Whenever a variable is defined, the current DP setting is stored with the variable, and a numeric value of that variable will always be printed with that number of decimal places. Thus the DP command can be used to create variables for different needs. The most common use is to include the command '>DP 0' at the start of a document to force all numeric variables to be printed as integers.

Example:
```
>AV "Unit cost? " price,"Quantity? " quantity,"Discount? " discount
>IF discount=""
>SV discount=0
>EI
>SV vat=15
>SV disccalc=1-discount/100 vatcalc=1+vat/100
>SV total=price*quantity*disccalc
>SV totincvat=total*vatcalc
>DP 0
>SV pound=totincvat[W1]
>SV pence=totincvat[W-1]
>DP 2
```

The above example illustrates a number of possible ways to manipulate and calculate numeric variables, including an example of obtaining the equivalent of INT(num) and FRAC(num). In this case the variable 'pound' will contain the integer and the variable 'pence' the fraction. The '>DP 0' command is needed so that 'pound' and 'pence' are displayed as integers.

Numeric arguments can also be used for making comparisons and in this case '>', '<', '<>', '>=', '<=', and '=' can all be used.

Example:
```
>IF age>40; true if age is greater than 40
```

Variables can also be used as counters and markers in stored commands, to cause actions to happen after a certain number of times.

Example:

```
>SV counter=0
>RP
>AV description cost quantity
>SV total=cost*quantity  counter=counter+1
>-----------------.---------.-------------.-----------.----R
     &description&    &cost&    &quantity&    &total&
>UN counter=10
>CO  remainder of text follows
```

In the above example, the counter is set to zero at the beginning and a description, details of cost and the quantity are requested. 'SV' then calculates the total cost, assigns it to a variable called 'total' and increments the value of 'counter'. The next line prints the variables and 'UN' checks to see whether the value of 'counter' has reached ten. If not, then the loop is repeated, otherwise the loop will be exited and the remainder of the text printed.

Date and time

Special variables are provided to allow the system date and time to used. These are string variables which are updated whenever they are accessed. The following are provided:

SYS_DATE	the date, including name of month and year.
SYS_DAY	the name of the day.
SYS_LDATE	the date in long form (same as SYS_DATE).
SYS_MONTH	the number of the month.
SYS_SDATE	the date in short form as defined in CONFIG.
SYS_TIME	the time, in the form hh:mm:ss.

For example at midday on Friday 20 July 1990, the values returned would be:

SYS_DATE	"20 July 1990"
SYS_DAY	"Friday"
SYS_MONTH	"07"
SYS_SDATE	"20-07-1990"
SYS_TIME	"12:00:00"

Example:

To display the date in the form dd/mm/yy:

```
>SV year = SYS_DATE[w3]
>SV date = SYS_DATE[w1] + "/" + SYS_MONTH + "/" + year[3:4]

The date is &date&
```

Other system variables

These mainly return information about the environment under which Protext is running, and are provided for specialised uses. All variables return strings.

SYS_DOC	the name of the current document
SYS_VER	the Protext version number, e.g. "4.20"
SYS_PROG	the name of the program, e.g. "PROTEXT", "PROWORT"
SYS_LANG	the name of the language of Protext (in English). For example, the English version of Protext returns "ENGLISH", the German version (PROWORT) returns "GERMAN".
SYS_OSVER	the operating system version number
SYS_PAGE	the current page number

Self incrementing variables and paragraph numbering

A variable name ending with a full stop is automatically incremented after being substituted in the text.

Example:

```
 >SV k.=1
&k.&                ; this prints 1 and changes k. to 2
 >SV k.=k.-1        ; this does not auto-increment so does subtract 1
```

Letters may also be used, e.g.

```
 >SV a.="a"
&a.&                ; prints 'a' and changes a. to 'b'
```

Paragraph numbering

Self incrementing variables are useful for paragraph numbering.

To produce output in the following form:

```
1.  First main heading

    1.1         First subheading
    1.2         Second subheading

2.  Second main heading

    2.1         ...
```

Use

```
>SV a=1               ; variable used for main heading numbers
>SV b.=1              ; variable used for subheading numbers
&a.&.           First main heading

    &a&.&b.&        First subheading
    &a&.&b.&        Second subheading

>SV a = a + 1  b. = 1
&a&.            Second main heading

    &a&.&b.&        ...
```

For convenience define '&a&.&b.&<TAB>' as a macro and then just press the macro key at the start of each new section. The order of the main sections can be moved at will without the need to manually renumber the sections.

Roman numerals

The >DP command provides for numeric variables to be displayed as Roman numerals. This can be used in conjunction with self incrementing variables to give Roman numbering of paragraphs. The permitted range is I to MMMCMXCIX (1 to 3999).

```
>DP r      ; subsequently defined variables printed as lower case roman
>DP R      ; subsequently defined variables printed as upper case roman
```

27. Configuration

CONFIG is the program used to create configuration files for use with Protext. When Protext is first installed, a file called PROTEXT.CFG is created. A large number of Protext features may be individually configured; from the various colours used on screen, to the names of dictionaries to be used when checking spelling and the dimensions of the paper to be printed on.

A number of different CFG files may be created and saved with different names and the '.CFG' suffix. These may be loaded into Protext as and when needed, with the LOADCFG command, or using the 'Command line options' (See below) when loading Protext.

Note: Hard disc users, in particular, will find that it is convenient to save CFG files in different directories.

Configuration file search order

Unless a 'CFG' environment variable has been defined (see below), Protext will initially will search on the current directory for a PROTEXT.CFG file, then for PROTEXT.CFG in the PROTEXT directory on the current drive.

Using a CFG environment variable

Sometimes it is useful to tell Protext to look for the CFG file in a particular directory, especially on systems with more than one hard disc. If an environment variable 'CFG' is defined, Protext will take its value as the path on which to look for 'PROTEXT.CFG'. If Protext does not find a CFG file in this way it reverts to the search procedure as above.

PC *To define the environment variable use the 'SET' command in the 'AUTOEXEC.BAT' file in the root directory. For example:*

```
SET CFG=C:\PROTEXT\
```

Note that if the environment space becomes full it can be increased using the 'SHELL' command in CONFIG.SYS. See the MS-DOS documentation.

ST *If the environment variable 'CFG' is set, Protext will use it. Additional software such as a shell program or desktop replacement is required.*

Amiga *To define the environment variable use the AmigaDOS 'ASSIGN' command. For example (note the colon after 'CFG'):*

```
ASSIGN CFG: DH1:PROTEXT/
```

Archimedes *Define a system variable as follows:*

```
*SET CFG :4.$.PROTEXT
```

Floppy disc users: The CONFIG program is normally on the Protext Program disc, and so this disc should be inserted into drive A (*Amiga: df0:*) before using the command, to ensure that the configuration file is saved onto the correct disc.

Using CONFIG

CONFIG is a fully menu-driven program and is largely self explanatory. Messages on the screen give relevant help information. The menus show the current state of each option.

Menu options may be selected by moving a highlighted bar up and down using the ↑ and ↓ keys. When a bar is highlighted, pressing RETURN, ← or → will either change the option or call a submenu. Alternatively, each menu option is identified by a letter and pressing this letter will move the bar directly to that option.

There are 4 main types of options that are altered in different ways:

(i) Numbers. Pressing RETURN will open a window at the bottom of the screen allowing a number to be typed in. When editing pressing ESC will clear the text entered so far, pressing ESC again will cancel the entry and restore the original. Pressing RETURN will confirm the entry, which will then be updated on the menu.

(ii) Strings. These options are edited in the same way as numbers. In some cases the strings are allowed to be longer than the space available to display them in the menu. They are then displayed in an abbreviated form, but pressing RETURN shows the whole string in the editing window.

(iii) Set of values. Some options are allowed to take a limited set of values, for example the keyboard language can be set to those languages that are provided. In these cases an editing window is not used. Pressing RETURN or → will advance the option to the next setting in the list, pressing ← will change it back. In this way it is very easy to cycle through all the allowed values. This manual does not list all the values because they may be changed from time to time.

(iv) On/off. These options are simply set to 'on' or 'off'. Effectively this is a simple case of (iii). Yes/no options are similar.

The initial menu contains the following options:

I - Initial Installation

Those options which may be important when first installing Protext are collected together in this submenu.

M - Main configuration menu

Calls a further menu allowing access to all configuration options via various submenus.

S - Save new configuration

This is a very important option which must be used when any changes are made. The name of the configuration file will be displayed when 'S' is pressed and usually it will simply be necessary to press RETURN to save the CFG file. It is possible to change the name at this point to create a different file.

Note that the current configuration file name is always displayed at the top of the screen when using CONFIG.

L - Load configuration

When CONFIG is used it will automatically load the default CFG file (following the same search procedure that Protext uses). This option may be used to edit different CFG files.

R - Reset default configuration

This resets all values to the Protext default settings.

Q - Quit

Quits CONFIG and returns to Protext or the operating system. If changes have been made but not saved a warning message will be given.

The options available are listed below together with their default values. In most cases there is a Protext command (edit mode, command mode or stored command) or a command line option to change the setting and these are listed too. Full details of the functions can be found by referring to the section of the manual covering the related command. In some cases further notes are given here.

There may be one or two variations from the list below as some options do not apply to all computers and new options may be added from time to time.

Configuration options

Option	Default	Related commands	Note
Editing options			
Insert or overwrite mode	INSERT	CTRL-TAB	
Right justification	ON	CTRL-J	
Word wrap	ON	CTRL-W	
Tabs & returns displayed	OFF	CTRL-V T	
Spaces displayed	OFF	CTRL-V S	
Control codes displayed	OFF	CTRL-V V	
Page mode	ON	CTRL-P	
Page break display	ON	CTRL-V P	
Auto re-formatting	ON	SHIFT-CTRL-R	
Layout options			
Page length	66	>PL	
Line spacing	1	>LS	
Top margin	3	>TM	
Header margin	2	>HM	
Footer margin	2	>FM	
Bottom margin	3	>BM	
Side margin	5	>SM	
Continuous printing	ON	>CP	

New page after printing	ON	>NP	
Headers	OFF	>HE	
Footers	OFF	>FO	
Edit default Ruler line	70 cols	>--	

Spelling checker options

Spell checking language	UK English	>SL	
Dictionary 1	ARNORENG.LEX	-	0
Dictionary 2	ARNORE2.LEX	-	0
Dictionary 3	USER.LEX	-	0
Dictionary 4	-	-	0
Dictionary 5	-	-	0
Quick dictionary	ARNORENG.QIC	-	0
Update dictionary	USER.LEX	-	0
Dictionary prompt	OFF	-	
Spell check whilst typing	OFF	SHIFT-CTRL-S	

File and disc options

Search path	A:\	-	1
Temporary text path	A:\	-	2
Default Extension to load	none	EXT	
Document mode extensions	various	-	3
Program mode extensions	various	-	4
AutoeXec file name	EXFILE	/X	0
Use GEM/ARP file selector	*NO*	-	5
Create icons for data files	YES	-	6
Automatic save time period	900 (secs)	-	
Automatic save maximum size	10240	-	

Memory options

File 1 memory size	0	/M	7
File 2 memory size	0	/M	7
Maximum file memory size	0	-	7
Spell memory	40000	/S	8
Ignored words buffer size	2048		
Undelete buffer size	2048	-	

Printer driver options

Default printer port	*PAR1*	*PAR,SER*	
PAR1 printer driver	-	*PRINTER*	*9*
PAR2 printer driver	-	*PRINTER*	*9*
PAR3 printer driver	-	*PRINTER*	*9*
SER1 printer driver	-	*PRINTER*	*9*
SER2 printer driver	-	*PRINTER*	*9*
Serial delay	*1000*	*/D*	*10*
Ignore Reset code	NO	-	
Ignore Line feeds	NO	-	
Form feeds	OFF	>FF	

Keyboard options

Protext Keyboard driver	*ON*	*/K*	*11*
Key delay (18th secs)	*10*	-	
Key repeat period (18th secs)	*1*	-	
Keyboard language	see note	LANG	12
Keyboard type	see note	KEYB	13
Use WordStar keystrokes	NO	/W	14
Standard keyboard driver	*YES*	-	*15*
CTRL/ALT/SHIFT lock	NO	/L	16
Mouse on/off status	*ON*	-	*17*
Redefine keyboard layout	-	KEYDEF	18

Display options

Set Colours	-	*INK (ST)*	19
Screen Mode	*Use initial mode*	*ALT-F,ALT-G*	
Edit mode cursor type	*Full height*	-	
Command mode cursor type	*Variable height*	-	
Cursor flash period	1/3 sec	-	
Screen blink time (ST)	*0 (off)*		*20*
Mono background colour (ST)	*BLACK*	*INK*	
Scroll lock forced on (PC)	*NO*	-	
Screen Fix (PC)	*OFF*	-	*21*
Text mode symbol defns. (PC)	*OFF*	-	*26*
Text font name (Amiga)	-	-	*22*
Text font height (Amiga)	-	-	*22*

Other options

Document or program Mode	DOC	DOC,PROG	
Program mode Tabs	*8	TAB	
Auto indent	OFF	SHIFT-CTRL-I	
Backups when saving	ON	-	23
Start in command mode	NO	-	
Sideways scrolling increment	1	-	
Decimal character	.	>DC	
Background printing	ON	BACK	
Audible warning disabled	NO	-	
Turn off clock	NO	-	
Date format	dd-mm-yy	-	24
Sorting method for accents	ANGLO-GERMAN	-	25

Notes about configuration options

0. A drive and path may be specified as part of the filename, so that a file in a particular directory will be used, rather than searching for the named file in the current and path directories.

1. The search path is used by Protext when looking for dictionaries, printer drivers, help files and external programs (CONFIG and CONVERT). Normally it is set up by selecting the disc configuration in the 'initial installation' menu.

2. The temporary path is used by Protext for storing temporary files when editing large files. Normally it is set up by selecting the disc configuration in the 'initial installation' menu. If a large RAM disc is available (*using expanded or extended memory on a PC*) enormous speed improvements can be achieved by setting the temporary path to use the ram disc.

3. When a file with one of these extensions is loaded, Protext switches to document mode. If such a file is printed Protext uses document mode printing. If two consecutive commas are typed when editing the list, it will cause files with no extensions to switch to document mode. Up to 15 extensions may be specified.

   Protext will also switch to document mode if a file with a stored command on the first line is loaded.

Amiga/Archimedes Note: These CONFIG options, though working, are redundant. This is because Protext documents are stored with carriage return codes at the end of each line and program files are not. This enables Protext to select the correct mode (document or program) depending on the file being loaded.

4. When a file with one of these extensions is loaded, Protext switches to program mode. If two consecutive commas are typed when editing the list, it will cause files with no extensions to switch to program mode. Up to 15 extensions may be specified.

5.

ST	*This option allows the GEM (or replacement) file selector to be used in Protext.*
Amiga	*The ARP file requester may be used if arp.library is present in the 'libs:' directory. If the library is not present this option will be ignored.*

6. *(Amiga only). See 'Getting Started - Amiga'.*

7. There are various options for allocating the amount of memory for text. The default, when all settings are zero, will usually be used. Protext will then allocate memory depending on the size of the files being edited. If values are entered for the file 1 and file 2 memory sizes then these fixed amounts will always be allocated. The file 2 setting is used for all files other than the first.

   The maximum file memory size can be used if the other values are zero. Then Protext will automatically allocate the size required for each file but will not exceed the maximum. This can be useful when editing a very large file to prevent that file being allocated too much memory and thus leaving some memory for other files or other uses.

8. This amount of memory is needed to perform the SPELL command. If this command gives 'out of memory' then reduce this figure.

9. To assign one of the listed printers to a port simply press RETURN, move the cursor to the required printer and press RETURN again. To assign a driver that is not listed press ESC when the list of drivers is on the screen, type the printer driver name and press RETURN. This manual entry method can also be used to supply a full pathname for the printer driver. Normally Protext will look for the printer driver on the search path.

ST note: there are only two printer ports - serial and parallel.
Amiga note: there is only a 'default printer driver' option. The printer port should be selected using Preferences.
Archimedes note: there is only a 'default printer driver' option. The printer port should be selected using The Configuration application from the Desktop.

10. In the unlikely event of a serial printer losing characters, increasing this figure may cure the problem. If not, then the printer cable is probably wired incorrectly.

11. *(PC only) Protext installs its own keyboard driver to replace the DOS keyboard driver. This may not work on some MS-DOS machines which are not fully IBM compatible. If the keyboard is not working correctly (or not at all) in Protext, set this to 'OFF'. See Appendix 3 for variations in editing commands caused by doing this.*

12. This option should be used to select the correct keyboard layout.

    PC Note: When Protext is first installed on a hard disc, or when 'reset configuration' is used, the keyboard language will be set to the appropriate language for the country code specified in CONFIG.SYS. This means that it will usually produce the correct language for the keyboard.

13.

PC *There are three keyboard types provided. The default is the standard XT/AT keyboard layout. This works on many keyboards irrespective of the physical position of the keys. The 'extended AT' keyboard should be selected when using a 101 or 102 key keyboard (with function keys at the top and separate cursor key cluster). The 'alternate AT' keyboard should be selected if it is found that the '\ ¦' and '# ~' keys have been reversed when using Protext.*

ST/Amiga *The first layout is the standard Protext layout. The second is the same except that the numeric keypad emulates the function of the keypad on a PC. On an Amiga this means that the keys operate according to the legends on the front of the keys. These legends are not marked on an ST keyboard but are as follows:*

7 Home	*8 ↑*	*9 PgUp*
4 ←	*5 (blank)*	*6 →*
1 End	*2 ↓*	*3 PgDn*
0 Ins		*. Del*

The keypad + and - keys will now perform the next and previous find functions, as on a PC.

On the Amiga a third keyboard layout is provided which follows the Amiga Intuition standard. See 'Getting Started - Amiga'.

14. The WordStar keystrokes provided are documented in an Appendix.

15. *(PC only) It may be necessary to turn this option OFF when using certain keyboard enhancer programs. We have not come across such a program yet, however. The symptom that would indicate that this option should be turned off would be where problems are experienced with the keyboard after leaving Protext. The effect of turning this off would be that keys typed ahead before leaving Protext will be lost.*

16. This option causes Protext to be entered in 'sticky key' mode. If CTRL, ALT or SHIFT is pressed the effect will carry forward to the next key pressed. It is, therefore, unlike Caps Lock in that the shift state is locked for the next key only. Multiple shift states may be used by pressing all the keys separately.

    The benefit of this is that a key combination such as SHIFT-ALT-A (which would normally require SHIFT and ALT to be held down while A was pressed) may be entered by pressing SHIFT, then ALT, then A. This will be of particular use to those with certain physical disabilities.

17. *(Not Amiga). This option has three settings: 'OFF' means the mouse may not be used, 'ON' means the mouse pointer is always visible in edit mode, 'HIDDEN UNTIL USED' means the mouse pointer is initially not visible and becomes visible when the mouse is moved. The default is 'ON'.*

18. Each key on the keyboard (except shift and lock keys) can be configured to produce a specified 2 byte code (token) for each of the following: by itself, with SHIFT, with CTRL, with ALT, with NUM LOCK on, with CAPS LOCK on, with SHIFT and ALT, with SHIFT and CTRL.

    When this option is selected, press the required key. The current definition will be listed. Each code is shown in 4 forms: hexadecimal, decimal, ASCII and command description. The last item describes the effect of command keys when used in Protext. If the key has been redefined previously the message 'REDEFINED IN CFG FILE' will be displayed. If not the values shown are the defaults for the currently selected keyboard type and language. To edit a definition (or create a new one) press 'E'. The key codes may then be edited in hex, decimal or ASCII (press TAB to change between them). To remove a definition from the CFG file press 'C'.

19. The colours can be changed for various different parts of the Protext screen. A visual representation of the colours makes this option very easy to use.

20. *(ST only). This option controls the speed of flashing the screen which replaces the error beep since there is no built in speaker.*

21. *(PC only). This may need to be set to 'ON' on certain computers if there is interference ('snow') on the display when the screen is re-drawn. This does not seem to be necessary on any new computers.*

22. *(Amiga only). 'Diskfont.library' must be available to use a different font. If using floppy discs this should be copied onto the startup disc.*

23. Protext normally creates a backup file, with the '.BAK' suffix, from the previous copy of the file being saved. This feature can be turned off in order to save disc space if using a floppy disc system.

24. This option allows the date format used by DIR and (*on the ST*) DATE to be set either to the European format (dd-mm-yy) or the U.S. format (mm-dd-yy), or the Swedish format (yy-mm-dd).

    PC Note: This date format will default according to the COUNTRY code defined in CONFIG.SYS unless explicitly set in CONFIG.

25. This option should be set by Scandinavian users to ensure correct index sorting.

26. *(PC only) Set this to ON to allow use of SYMBOL and the definitions of characters ø Ø ł Ł ŀ Ŀ ı ð Đ þ and Þ in text mode on EGA/VGA. This is normally disabled because defining symbols causes the screen to flicker.*

Command line options

Some of the configuration options are also available as command line options which may be typed when Protext is run. A command line option will always take precedence, so this provides an easy way to override some of the settings just for one occasion. Any number of options may be used together.

Command line options are provided as follows:

-C<fn>	causes specified CFG file to be used.
-Dn	inserts a delay during printing to a serial printer, the delay being 8n/r microseconds where r is the clock speed in Mhz. The default setting is 1000, which gives a millisecond on an 8Mhz machine.
-K	*(PC only) uses MS-DOS keyboard driver.*
-L	sets 'caslock' mode.
-Mn1,n2	allocates n1 bytes for the first document and n2 bytes for the second. If n2 is omitted both are set to n1. The minimum size in each case is 10000.
-Sn	allocates n bytes for spell checking text from disc.
-W	uses WordStar keystrokes.
-X<fn>	execs the specified file.
-Wx,y,w,h	*(Amiga only) defines the size and position of the Protext window. x,y is the position of the top left corner, in pixels. w and h are the window dimensions, in pixels.*

ST/Amiga *The command line options may be typed after the command name when running Protext from a shell or CLI. Otherwise it is possible to use options by saving the line in a file called 'PROTEXT.OPT' in the same directory as the Protext program.*

System Configuration Options

Amiga *Protext uses and abides by all the settings made with Preferences, with the exception of printer drivers. The following Preferences facilities are relevant to Protext:*

Colour settings
Keyboard repeat speed and delay
Printer port (parallel or serial)
Serial port options (if using serial printer)
Mouse speed; Double click delay; Mouse pointer shape
Time and date

For explanations of these options and how to change them see the documentation supplied with the Amiga. Preferences is supplied on the Protext disc and any changes that are made using Preferences after booting up with the Protext disc will be saved onto that disc.

ST *Protext uses and abides by all the settings made with the Control Panel from the GEM 'Desk' menu with the exception of the 'Install Printer' options. The following settings are relevant to Protext:*

Colour settings
Keyboard repeat speed and delay
Serial port options (if using serial printer)
Mouse speed
Time and date
Keyboard click on/off; Bell on/off

For explanations of these options and how to change them see the documentation supplied with the ST.

Archimedes *Most system settings are used by Protext including:*

Colour settings
Keyboard repeat speed and delay
Printer port (parallel or serial)
Serial port options (if using serial printer)
Mouse speed; Double click delay;

*Many of these options can be set either with *CONFIGURE or using the Configuration application from the Desktop.*

28. File Sorting

FSORT is a separate program which will sort the contents of a file any of several ways. It was designed with the specific requirements of sorting datafiles containing names and addresses, but is of more general use.

FSORT can be used either from Protext command mode or outside Protext.

Running FSORT from Protext command mode

Ensure that the FSORT program is on the disc in the current drive or on the search path and type:

```
FSORT <input filename> <output filename> <options>
```

Running FSORT as a stand alone program

FSORT is a very useful program in its own right, and may be used independently from Protext. This may be necessary if a message appears indicating that there is insufficient memory. On an IBM PC or compatible it can be used in the same way from the DOS command prompt. On an Atari ST it can be used from the GEM desktop by double clicking on FSORT.PRG and typing the filename and options in response to the prompt. On an Amiga it can be used from the Workbench by double clicking on the FSORT icon, or by typing 'FSORT' from the command line interpreter.

Sorting options

(i) Options that define the type of file record

-Fn	fixed record of length n
-V	variable record length
-T	fields delimited by tab

The first thing to note is whether the data is stored with fixed or variable length records.

Variable record files:

Use the option: -V

Fixed record files:

Use the option: -Fn

Replace n by the number of fields in a record, so for a file with records of 7 lines, use '-F7'. If a blank line has been left at the end of each record it must be counted in the number of lines.

Default option:

If neither '-F' nor '-V' are used, FSORT will assume variable record files.

(ii) Option to specify the field or fields on which to sort

-Sa,b,c,... sort on fields a,b,c,...

Here, 'a', 'b', and 'c' will be replaced by field numbers.

Unless otherwise specified, the records will be sorted using the first field. If a sort using a different field is required, the 'S' option should be used. This works in the same way whether fixed or variable record files are used.

To sort on the second field use '-S2'.

If a negative field number is used, it means count the fields backwards from the end of the record. For example, to sort on the last field in a variable record file, use '-S-1'.

Secondary sort fields can also be specified, to be used to order records when the first sort field is identical. For example, to sort on the penultimate field and if that is the same on the third field, use '-S-2,3'.

The field numbers should be separated by commas with no intervening spaces. Any number of sort fields may be specified, e.g. '-S1,2,3,4,5'.

Default option: If '-S' is not used, FSORT will assume '-S1'.

(iii) Options that define the method of sorting

-A	alphabetic sort
-Z	reverse alphabetic sort
-P	personal name sort
-W	sort on first word
-L	sort on last word
-N	numeric sort
-D	decreasing numeric sort
-M	set sorting method (0 = English/German, 1 = Scandinavian)

There are many different ways of sorting provided which cater for a wide range of applications. These options may all have field numbers specified as parameters (see below). The options are:

A - Simple alphabetic sort
Z - Reverse alphabetic sort

These are straightforward alphabetic comparisons.

P - Personal name sort

This takes the last word on a line and puts it at the start before comparing. This is the sort method needed for sorting individual names. For example when asked to compare 'Alan J Brown' and 'J W Bell' it will switch them round and compare 'Brown Alan J' with 'Bell J W'.

Often, names of organisations will be mixed with individual names, e.g. 'Arnor Ltd'. If the method described above was used this would be sorted as 'Ltd Arnor' which is obviously not intended. To avoid this FSORT recognises a convention to say 'sort this field using simple sort'. This is indicated by putting a marker at the start of the line in the datafile by typing CTRL-@ ?.

W - First word

This is similar to simple sort, but if the first words in the fields are the same it considers the fields to be equal and doesn't look any further. In other words it cuts off the fields being compared at the first space. This is mainly useful in conjunction with the 'Q' option.

L - Last word

Similar to first word sorting, this compares just the last word on a line.

N - Numeric sort
B - Backwards numeric sort

These are self explanatory. The number at the start of the field is used and any subsequent text is ignored. If the field being sorted does not start with a number it is sorted as if it contained zero.

Using different sort methods for different fields:

If secondary sort fields are specified it is possible to use a different sort method for each one. If one of the options 'A', 'Z', 'P', 'L', 'W', 'N' or 'D' is used with no following number, then all fields will be sorted using that method. Otherwise, numbers may be given with these options to specify the fields that use that type of sort.

For example, to sort a variable record address file using a personal name sort on field 1 with the last word of the last field (the postcode?) as the second sort field, use '-V -S1,-1 -P1 -L-1'.

(iv) Miscellaneous options

-T	sort tabulated columns
-H	list options
-Qn	query if n fields match

Help option

Typing either of the following will list the available options:

```
FSORT -H
FSORT ?
```

Sorting tabulated columns

The 'T' option causes FSORT to take a tab character as a field delimiter. Otherwise, fields are only delimited by the end of a line. This option allows files containing tabulated columns to be sorted using any column. Each record occupies exactly one line.

Using accented characters

All the characters listed in the 'Character Set' appendices will be correctly sorted and can be freely used.

Checking for duplicates

Taking the example of an address file, as used in the tutorial, it is very useful to be able to check that records being entered are not already present, so an option is provided to give assistance with this. It is slightly tricky because the same name may be typed in slightly different ways and so will appear different.

The 'Q' option will query any pair of records, which appear to be the same. The criterion for 'appear to be the same' depends on the sorting options chosen, as we shall see. When two records are compared during the sort process and they are sufficiently similar, the two records will be displayed on the screen. Either one can be then be deleted by typing '1' or '2', or they can both be left (type RETURN).

So what is 'sufficiently similar'? In the simplest case, it means that the first sort fields are exactly the same according to the selected sort method. For example, using the options '-S2 -L2 -Q', any pair of records with the last word on the second line being the same, will be queried.

The 'Q' option may take a parameter to extend this. If '-Q2' is used, the first sort fields must be the same and also the secondary sort field. For example, using the options '-S1,-2 -L1 -W-2 -Q2', a pair of records will be queried if the last words on the first lines are the same and the first words on the penultimate lines.

This is where the options for sorting on first or last word are very useful for sifting out duplicated entries which have been entered slightly differently, e.g. an extra initial or space. The records are displayed on screen allowing a visual check.

This test for duplication is very much a heuristic process, and there is no guarantee that all duplicates will be found. It is possible that certain records that would match will not be found, or that some matches will be queried more than once. The reason for this is that it depends on the sort algorithm - duplicates will only be queried when the sort algorithm compares them. Comparing every pair of records would be so inefficient that it cannot be considered. If this limitation is borne in mind, the 'Q' option is a very useful, practical means for checking for duplicate data.

Note: When using the 'Q' option, the file is not guaranteed to be sorted correctly for the reasons just discussed. After removing duplicates he file should always be re-sorted without the 'Q' option.

Some examples of using FSORT

(i) To sort a file called 'WORDS' which consists of a list of words, one per line.

```
    FSORT WORDS -F1 -S1
or  FSORT WORDS
```

(ii) To sort a file called 'NAMES' which contains a list of names and addresses separated by a blank line after each address.

Thus,

```
    FSORT NAMES -V -P       will sort by name, surname first
    FSORT NAMES -V -S2      will sort by the first address line
```

(iii) To sort a file called 'DATA' containing a set of data items, each 7 lines long, sorting by the last word on the fifth line:

```
    FSORT DATA -F7 -S5 -L
```

(iv) To sort a file called 'ADDRESS' into a new file called 'NEWADDRS', sorting by postcode (last word on the last line), followed by name if the postcode is the same and query any pair of records with the same postcode to allow duplicates to be removed:

```
    FSORT ADDRESS NEWADDRS -V -S-1,1 -L-1 -P1 -Q1
```

29. File Conversion

CONVERT is primarily provided for the purpose of converting text and data files created by other programs into a form more suitable for use by Protext. It may also be used to convert Protext files into forms suitable for use with other software.

Using CONVERT from the menu

To run simply type the command:

```
        CONVERT

or      CV
```

from Protext command mode. If there is insufficient memory to run the program it may be run from outside Protext either by typing 'CONVERT' from the DOS command line or Amiga/ST shell program, or by double clicking on the Protext icon from Gem desktop/Workbench.

Simply select the required conversion from the menu and give the name of the file which is to be converted. An 'output filename' may be specified, in which case the original file will be unaltered. If the output filename is omitted (by just pressing RETURN) the new file will have the same name and the original will be renamed as a backup file.

Using CONVERT from the command line

CONVERT can also be used from the command line avoiding the menu altogether. This is useful as it allows the creation of EXEC files to perform conversions on a series of files. The command is used as follows:

```
CONVERT <convert-type> <input file> (<output file>)
```

<convert-type> is a short name describing the particular conversion option. The output filename is optional. The input filename may include wildcards. fpr example:

```
CONVERT TOAMIGA *.PPD          (not Amiga)
CONVERT FROMAMIGA #?.PPD       (Amiga)
```

CONVERT has a built in help option which will list all the options. To obtain the list, type 'CONVERT ?'

At the time of writing the following conversion options are provided:

name	from	to
FROMASC	- ASCII	Protext
FROMCPM	- CPM/CPC Protext	Protext
FROMAMIGA	- Amiga/Archimedes Protext	PC/ST Protext
FROMWS	- WordStar	Protext
FROMFW	- First Word (Plus)	Protext
TOASC	- Protext	ASCII
TOCPM	- Protext	CPM/CPC Protext
TOAMIGA	- PC/ST Protext	Amiga/Arch. Protext
TOWS	- Protext	WordStar
TOFW	- Protext	First Word

Notes:

1. No conversion is required between PC and ST Protext files.

2. No conversion is required to transfer files between v3, v4 and v5 Protext.

3. FROMAMIGA / TOAMIGA

Conversion between Amiga Protext and PC or ST Protext is not always required for documents written in English but is required if the pound sign (£), any accented characters, or other special symbols are used.

A program called Crossdos can be used to read and write IBM format 3½" discs on an Amiga or serial transfer may be used. This allows IBM format discs to be freely used within Protext.

The Archimedes uses the same character set as the Amiga.

4. TOASC - ASCII files for use with DTP programs

Certain desktop publishing programs require ASCII files in a form with no carriage returns at the end of lines, only at the end of paragraphs. This option converts a Protext document to this form.

The new file will be ready to load into the DTP program. Each paragraph will now occupy only one line in the file. It is not normally recommended to edit the file in this form because certain operations are significantly slower on very long lines. The command FORMAT will restore the paragraph layout (but not any control codes).

Technical note: the effect of this conversion is to remove all control codes, soft spaces and soft returns.

5. FROMFW / TOFW

The TOFW option is provided for use on the Atari ST with DTP programs which accept First Word format without losing certain formatting features. The following are converted by these options:

Soft and hard spaces, margin sizes, page length, ruler lines, header, footer, pica, elite, condensed, expanded, bold, italic, underline, superscript, subscript, page breaks, conditional page breaks, paragraph formatting (soft returns inserted).

6. FROMWS / TOWS

The WordStar conversion options are provided so that files can be transferred to and from WordStar and also for transferring files to other programs which also offer a WordStar conversion option (such as Word Perfect and most DTP software). The following features are converted:

The full IBM character set, soft and hard spaces, margin sizes, page length, header, footer, pica, elite, condensed, double strike, expanded, bold, italic, underline, superscript, subscript, page breaks, page number, comment lines, paragraph formatting (soft returns inserted). Also the WordStar custom printer codes Q, W, E and R are converted to the Protext control codes W, X, Y and Z respectively.

User Defined Conversions - CVT files

CONVERT has a number of methods of conversion built into the program, but it is also possible to define new conversion types by means of conversion files. These are simple programs and are stored in files with the extension '.CVT'.

The following is a list of the commands that are available in conversion files:

MAKE

Syntax: MAKE charsequence = newcharsequence

Description: Substitutes one sequence of characters with another. 'charsequence' may be a string of characters within quotation marks or a number in decimal or hexadecimal (hex preceded with '#'). Any number of strings or numbers may be linked together to create a charsequence. 'newcharsequence' may be constructed in exactly the same way and will be the sequence of characters that replace 'charsequence'.

Examples:

MAKE "abcde" = "vwxyz" abcde will be replaced with vwxyz.

MAKE 65,66 = 49,50 will replace the character sequence 'AB' with '12', the four numbers being the decimal values of the ASCII characters A, B, 1 and 2 respectively.

IGNORE

Syntax: IGNORE n

Description: Causes n bytes to be skipped over, and not written to the output file. This allows file headers to be ignored.

QUOTES

Syntax: QUOTES

Description: Causes all lines to be wrapped in quotation marks. Sometimes useful when converting files produced by database programs.

BLANKS

Syntax:	BLANKS
Description:	Causes a blank line to be inserted after each line. Sometimes useful when converting files produced by database programs.

REM

Syntax:	REM <string>
Description:	Allows comments to be added to conversion files.

WRITE

Syntax:	WRITE <string>
Description:	Displays a message on the screen before converting the file.

Using CONVERT with a CVT file

Once a suitable CVT file has been created, containing the commands and parameters required, it should be saved with a suitable name. The syntax for CONVERT is:-

```
CONVERT <cvtfile> <infile> <outfile>
```

'infile' is the file to be converted and 'outfile' is the name that the converted file is to be saved with. 'cvtfile' is the CVT file created as described above. If 'outfile' is omitted then 'infile' will be used for the new file, and the original file will be renamed with the extension '.BAK'.

If 'cvtfile' matches one of the built in conversion types, then this will be used. If a conversion file has the same name it may be specified by typing the '.CVT' extension to force the file to be used rather than the built in type.

It will often be most convenient to copy the CONVERT program and the required '.CVT' files onto the disc containing the text files to be converted.

30. Creating Printer Drivers

Warning: This chapter is difficult. Protext is supplied with a wide (and ever widening) range of printer drivers and the vast majority of users need know nothing of the contents of this chapter. It is provided for those wishing to create their own printer drivers and assumes an understanding of the means of controlling printers from software.

Printer drivers are files with the extension '.PPD' (Protext Printer Driver). To change or create a printer driver one of these files is edited - the codes are typed into the file using special commands which are detailed below. Usually it will be found possible to modify an existing printer source file, rather than start from the beginning.

This chapter contains a list of the commands that are available for use in printer drivers, followed by an explanation of how to use the more complex commands concerned with microspacing and proportional printing.

code sequence	may be one or more characters wrapped in quotes, or a number in decimal or hex (preceded with '#') or a mixture of both.
number	is an integer number.
code	is a single character (wrapped in single or double quotes) or its numeric value in decimal or hex (preceded with '#')

Each command must be on a line of its own. If the code sequences for a command are too long to fit on one line, a plus sign, '+', may be put at the end of the line immediately before a carriage return to indicate that the command continues on the following line.

The printer driver commands are divided into three sections for clarity:

(i) General commands
(ii) Accent definition commands
(iii) Font definition commands

A printer driver normally consists of various general and accent commands, which relate to all fonts, followed by zero or more font definition sections (marked by the FONT command).

(i) General commands

These commands should appear before the first FONT command to ensure that they are correctly defined for each font.

AB

Syntax: AB <code sequence>

Description: Abandon code sequence. These codes are sent to the printer when the ABANDON command is used.

CC

Syntax: CC <code> = <code sequence> (; <code sequence>)

Description: Define printer control code. <code> is the required printer control code letter ('a' to 'z' or '@'). The first item after 'CC' is the control code letter, followed by the sequence of codes for 'on', followed (optionally) by a semicolon and then the codes for 'off'. When the printer control code is used (by typing CTRL-X and the letter) the first code is translated to the 'on' sequence and the second to the 'off' sequence and from then it alternates. If no 'off' code is defined the 'on' sequence is always sent. The same as the 'CC' stored command.

Example: CC z 8 '^'
REM prints circumflex on letter

COPIES

Syntax: COPIES <code sequence> (; <code sequence>)

Description: Defines the control code sequence used to print multiple copies on a laser printer. When the multiple copy printing is enabled (by 'MP ON' in the printer driver or as a stored command) Protext will print multiple copies by sending this code sequence before each page. The number of copies required is sent as an ASCII string between the two code sequences. If the second code sequence is omitted the number is sent as a single binary byte after the code sequence.

CP

Syntax: CP <code sequence>

Description: Continuous printing code sequence. These codes are sent to the printer at the start of each page when continuous printing is selected.

DOUBLE

Syntax: DOUBLE <letters>

Description: This command is used to indicate that the control code letters specified cause all character widths (fixed pitch and proportional) to be doubled. This is to simplify and save space in the printer drivers.

The default (if no DOUBLE command is present) is DOUBLE L. DOUBLE with no following letters is used to disable this default setting.

END

Syntax: END <code sequence>

Description: The specified code sequence is sent at the end of every copy printed, and is useful for resetting the printer back to a default state. Compare: the reset code which is sent at the start of every copy.

FD

Syntax: FD <letters>

Description: This will set the default font for printing. This will be overridden by any '>fd' stored commands or control codes at the start of the text.

Example: FD rl

FF

Syntax: FF ON/OFF

Description: Form feeds on/off. The same as the 'FF' stored command.

FP

Syntax: FP ON/OFF

Description: Formatting whilst printing defaults to ON if 'PP ON' is used. 'FP OFF' may be used to change this.

GROUP

Syntax: GROUP <letters>

Description: The specified list of control code letters are linked in that whenever one of the group appears in the text, the others are automatically turned off. This is to eliminate the need to turn off one state before entering another. Currently only one group is supported.

There is an exception to this rule. When one of the control codes in the group occurs twice with no intervening control codes it works as a normal 'off' code and has no effect on the other codes in the group. Only one works, not nested codes.

Example: GROUP hrg

hr is treated as r (the r turns h off)
hrr is treated as h (the second r turns r off)
hrh is treated as h (the second h turns r off)

HL

Syntax: HL <code sequence>

Description: Half line feed code sequence. These codes are sent to the printer to advance the paper by half a line, when '>LS' is used to select half line spacing or >IP is used to change the inter-paragraph space.

MC

Syntax: MC <code sequence> (; <code sequence>)

Description: Microspacing code sequence. The same as the 'MC' stored command. If the general purpose microspacing method (MM 0) is used, Protext must be given a code sequence to print a microspace. This is the sequence of codes that moves the print head by a microspace. The way this is done will vary for different printers (and it may not always be possible). As an example the following definition could be used for Epson printers:

```
>MC 27 "L" 1 0 0
```

The codes 27, "L" put the printer into 'double density graphics bit image mode'. The next two bytes are the number of bytes of data to be printed, low byte first (i.e. 1 byte of data). The last zero is the one byte of data. The effect is that the print head is moved 1/120th of an inch to the right.

The second code sequence is used by the other general purpose microspacing method (MM 6), described below.

MENU

Syntax: MENU

Description: Starts the table defining the text and control codes for the font changes that are to appear in the Style menu. Lines following the 'MENU' line should contain a textual string in quotes and a control code or codes. The command 'MENUEND' is used to terminate the list.

Example:

```
MENU
"TMS RM 10 pt" r
"TMS RM 8 pt" rc
"Helve 8 pt" h
MENUEND
```

MENUEND

Syntax: MENUEND

Description: Ends the menu table (see 'MENU').

MF

Syntax: MF <number>

Description: Microspacing factor. The same as the 'MF' stored command. See below.

MM

Syntax: MM <number>

Description: Microspacing method. The same as the 'MM' stored command. The general method (MM 0) sends a code sequence for each microspace (MC must then be used). The special methods send a single code for each group of microspaces and give much faster printing on the appropriate printer. See below.

MP

Syntax: MP ON/OFF

Description: Used to select multiple printing by default. This is used on laser printers for fast printing of multiple copies - each page is only sent to the printer once and printed the required number of times. The same as the 'MP' stored command.

MS

Syntax: MS ON/OFF

Description: Used to select microspacing by default for printing.

OMIT

Syntax: OMIT <list of control code letters>

Description: This is used to omit attributes from the bottom part of the style menu for printer drivers where some of the standard attributes are not available.

Example: OMIT ceinpqst ; used in SIMPLEUK

PP

Syntax: PP ON/OFF

Description: Used to select proportional printing by default.

PSCODE

Syntax: PSCODE <code sequence> ; <code sequence>

Description: Paper sensing code sequence. If a '>PS ON' command is used, the first code sequence is sent to the printer at the start of each page. The second code sequence is sent to the printer after the header margin has been printed. Not all printers support these codes, but if they are available they may be used to turn the printer off-line after each page when background printing on single sheets.

RC

Syntax: RC <code> = <code sequence>

Description: Define character translation. The first item after 'RC' is the character to be redefined, this is followed by the sequence of codes to be printed whenever the character occurs. This is the same as the 'RC' stored command.

Examples:
RC £ 27 ">" 1 27 "="
RC "!" = "!" 8 "."

Notes: 1. RC will accept accented characters that are not in the standard character set (and are stored as letter+accent), for example (on a PC or ST) 'RC È=163'.

2. All accented characters, including those in the standard character set will be split into the letter and accent for printing, unless an RC definition exists for the character or the appropriate accent command is not present. This means (**change from v4**) that even when the printer is using the same character set as the printer an RC must be used (e.g. RC ä = ä).

REGULAR

Syntax: REGULAR <letters>

Description: This command specifies a list of control code letters that can be turned off and back on again in any order and still return to the same printer state. This is used so that header and footer text can be printed in the correct style.

REM

Syntax: REM remark

Description: Any text following a REM command on the same line will be ignored. This command is for the purpose of adding explanatory comments to a printer driver.

SIMPLE

Syntax: SIMPLE ON

Description: This would be used if a printer does not support bold and underlining. These functions will then be carried out by backspacing and over-printing.

SS

Syntax: SS <code sequence>

Description: Single sheet code sequence. These codes are sent to the printer at the start of each page when printing on single sheets.

(ii) Accent definition commands

ACUTE	<prefix sequence> ; <postfix sequence>
BREVE	<prefix sequence> ; <postfix sequence>
CARON	<prefix sequence> ; <postfix sequence>
CEDILLA	<prefix sequence> ; <postfix sequence>
CIRCUMFLEX	<prefix sequence> ; <postfix sequence>
DACUTE	<prefix sequence> ; <postfix sequence>
DGRAVE	<prefix sequence> ; <postfix sequence>
DIAERESIS	<prefix sequence> ; <postfix sequence>
GRAVE	<prefix sequence> ; <postfix sequence>

MACRON	<prefix sequence> ; <postfix sequence>
OGONEK	<prefix sequence> ; <postfix sequence>
OVERDOT	<prefix sequence> ; <postfix sequence>
RING	<prefix sequence> ; <postfix sequence>
TILDE	<prefix sequence> ; <postfix sequence>

This group of commands deal with the printing of accents (or diacritical marks) above letters. This method of printing is used for those accented characters that are not provided in the printer as a single character (as, for example, ä and è are).

The usual method is to print the letter, then to backspace and print the accent. There is a modification to this when the letter is upper case - then the accent needs to be printed at a higher position on the paper, so a code is sent before the backspace to raise the printing position and a code to lower it is sent after the accent.

In some cases the 'letter-backspace-accent' sequence may not be sufficient. Therefore the possibility of printing a code sequence before the letter is also provided. A possible example of use would be to change symbol set before printing the letter.

Each accent command is followed by two code sequences, separated by a semi-colon. The first sequence is sent before the letter, the second is sent after the letter. The most common case of 'letter-backspace-accent' has an empty first sequence. For example:

```
acute     ; 27,"&f0S",27,"(8U",8,168,27,"(10U",27,"&f1S"
```

Here, the code sequence after the semi-colon prints a backspace and an acute accent.

ASCEND

Syntax:	ASCEND <up sequence> ; <down sequence>
Description:	ASCEND is used to define the codes necessary to raise and lower the printing position. This is used when printing accents on upper case letters, as desribed above.
Example:	ascend 27,"j",5 ; 27,"J",5

(iii) Font definition commands

These commands should appear within a font definition section. If they are used before the first FONT command they are applied to font 0.

COPYCC

Syntax: COPYCC <font number>

Description: This works in the same way as COPYRC but for CC definitions. CC commands (as above) can be used within a font definition section.

COPYRC

Syntax: COPYRC <font number>

Description: This causes the RC translations for a font to be initialised to those of the specified font number. The default if no COPYRC command is present is font 0. RC commands may be used after the COPYRC line to modify these translations.

CW

Syntax: CW <number>

Description: Defines the character width in granules for a fixed pitch font. This is equivalent to setting PW to this value and WIDTH to the same value for every character. The default setting is 12 (10 cpi with 1/120th inch granule). See below.

Note: The >CW stored command is ignored in Protext version 5 and above.

ENTER

Syntax: <code sequence>

Description: Defines the code sequence sent when the font is entered.

EXIT

Syntax: <code sequence>

Description: Defines the code sequence sent when the font is left.

FONT

Syntax: FONT <number>

Description: Introduces a font definition. A font definition is terminated by another font command or the end of the file.

PW

Syntax: PW <number>

Description: Defines the 'average' proportional character width in granules. This is the same as the 'PW' stored command. See below.

RULES

Syntax: RULES <letters> = <fontnum>{,<letters> = <fontnum>}

Description: The RULES command is used to define the control codes that are to be used to access each font. The letters are the usual Protext printer control code letters. The order of the rules is critical as a 'first match' principle is used.

Example: rules ce=3,c=2,e=1

These rules are used to determine the printing font as follows:

If c and e are enabled (condensed elite) then font 3 is used. Otherwise if c is enabled font 2 is used, else if e is enabled font 1 is used, else font 0 (the default) is used.

WIDTH

Syntax: WIDTH <code> = <number>(,number)(,number)...
W <code> = <number>(,number)(,number)...

Description: Specify width of a character or characters for proportional printing. Further widths may be entered for following characters on the same line. See below.

Example: WIDTH 'A'=32,28,29,33

is equivalent to:

WIDTH 'A'=32
WIDTH 'B'=28
WIDTH 'C'=29
WIDTH 'D'=33

WIDTHS

Syntax: WIDTHS

Description: Used to define a complete table of character widths. The lines following the 'WIDTHS' command contain characters and widths in the following form:

```
Ç29 ü22 é20 â20 ä20 à20 å20 ç19 ê20 ë20 è20
```

This form is recommended as it is both easier to read and occupies less space on the disc.

The end of the table is automatically detected.

Note: Each width table takes up ½K of memory when the printer driver is loaded.

Defining fonts

A printer driver may contain up to 64 different font definitions. Each font is identified in the driver by a number. This number is only used within the printer driver. Access to a font in the text is obtained by using an appropriate printer control code or combination of codes.

The font selection rules operate in addition to the usual printing of any control code sequences defined by 'cc'. In some cases there might be no associated code sequence because each font has a defined 'enter' and 'exit' code sequence which is printed when the font is selected.

The information required to define a font is:

(i) enter sequence - codes printed when changing to the font
(ii) exit sequence - codes printed when changing from the font to another
In simple cases these sequences will not be needed as the font changing is performed by the control code sequences.

(iii) For a proportional font: The character width table and the average proportional width (PW).

For a fixed pitch font: The character width (CW).

The default printing font

The default printing font determines the relationship between ruler lines and the width of the printed text and the position at which tabulated text is printed. It also determines the side margin width. These are all calculated using the PW value. See 'Proportional printing'.

The '>FD' stored command will change the ruler line and tabs as per the new PW value, but will not affect the side margin width. The side margin will always be calculated using the PW of the initial default font.

CPD files

When Protext loads a printer driver it will sometimes create a new file with the extension 'CPD' (Compressed Printer Driver). As the name suggests this is a compressed form of the 'PPD' file and is used to speed up the loading of the printer driver. The use of 'CPD' files is completely transparent and they are mentioned here only to explain their presence on the disc. If a 'CPD' file is deleted or the corresponding 'PPD' is edited then Protext will automatically re-read the 'PPD' file next time the driver is loaded and create a new 'CPD'.

Note: A 'CPD' file cannot be converted back to a 'PPD' file, and therefore the original 'PPD' printer driver file must not be deleted.

Portability and Compatibility with Protext version 4.3

Although Protext version 4.3 supports a subset of the printer driver commands it is perfectly possible to create printer drivers that work with v4.3 and v5. All the drivers supplied by Arnor are of this form.

The commands not supported by v4.3 are:

ASCEND, COPYCC, COPIES, COPYRC, DOUBLE, END, ENTER, EXIT, FD, FONT, GROUP, MENU, MENUEND, MP, OMIT, REGULAR, RULES

and the accent commands:

ACUTE, BREVE, CARON, CEDILLA, CIRCUMFLEX, DACUTE, DGRAVE, DIAERESIS, GRAVE, MACRON, OGONEK, OVERDOT, RING, TILDE

The three commands IF, ELSE and ENDIF are used to enclose blocks containing commands that are only supported by version 5.

IF
Syntax: IF <identifier>

ELSE
Syntax: ELSE

ENDIF
Syntax: ENDIF

Descriptions: Each version of Protext has a defined list of identifiers. If any of these are used in an IF statement then the lines between the IF and the corresponding ELSE or ENDIF are processed. If anything other than one of the list of defined identifiers appears then these lines are not processed but any lines between the corresponding ELSE and ENDIF are processed. IFs may be nested.

Protext version 5 and above has the identifier 'VER5' defined. This can be seen in use in most of the supplied printer drivers as in:

```
IF VER5
... lines here use commands specific to version 5 ...
ENDIF
```

Each version of Protext also has an identifier that indicates the type of computer and the character set. These are:

computer	identifiers	
PC	PC	IBM
Atari ST	ST	IBM
Amiga	AMIGA	ANSI
Archimedes	ARC	ANSI

Versions of Protext in other languages also have the language name defined (in English), for example 'SWEDISH' and 'GERMAN'.

Setting up a printer driver for microspacing and proportional printing

This section explains how to set up a character width table for a font using the commands WIDTH, MF, MM, CW and PW. Note that MF and MM are global and apply to all fonts, the other commands will be used within each font section. If WIDTH, CW and PW are used before the first font section they will be taken as applying to font 0, thus ensuring compatibility with pre version 5 printer drivers.

Definitions:

A **microspace** is the smallest space that Protext will insert when justifying. This is often the same as the smallest distance the printer can move the print head.

A **granule** is the unit in which character widths are measured. It is often the smallest possible difference between widths of proportional characters on a printer. A granule is usually chosen to be the same size as a microspace, except in cases where it is not possible to move the print head by a granule. Then a granule may be chosen as a smaller unit so that the characters may be measured in convenient units.

The **microspacing factor (MF)** is the microspace width divided by the granule width. This is used by Protext to define a granule. This is always a positive integer because the microspace is never smaller than the granule and the microspace size is always an integral multiple of the granule size.

Example: MF * granule = microspace (gpsi = granules per 60th inch)

	microspace	granule	MF	gpsi
Laser printers	1/300th inch	1/300th inch	1	5
Daisywheel printers	1/120th inch	1/120th inch	1	2
24 pin dot matrix	1/180th inch	1/360th inch	2	6
9 pin dot matrix	1/60th inch	1/120th inch	2	2
9 pin dot matrix	1/60th inch	1/240th inch	4	4

First it is necessary to choose the microspacing method (MM). Each method prints microspaces in a different way. Those available are:

MM 0	General method. Uses the code sequence defined with MC to print a microspace.
MM 1	Old Epson 9 pin. Uses the 'ESC K' code to print microspaces.
MM 2	HP Laserjet Plus. Uses the 'ESC *p+<#>X' code.
MM 3	Diablo daisy wheel printers. Uses 'ESC <31>'. Proportional mode is disabled while the microspaces are printed.
MM 4	Epson 24 pin, laser, new Epson 9 pin. Uses 'ESC \'.
MM 5	HP Laserjet. Uses the 'ESC &a+<#>H' code. 'MM 2' is preferred when this is available.
MM 6	General method. Uses two code sequences defined with MC, outputs the number of microspaces between the two sequences in ASCII form.

Method 0 should only be used if none of the other methods are suitable. This is because the other methods send a single code for each inter-word gap, whereas method 0 prints individual microspaces and so is slower. If method 0 is to be used the microspace code sequence must be defined. See 'MC', above.

The granule will normally be chosen from the table above. The corresponding microspacing factor (from the table) should be put into the printer driver with 'MF' in order to define the granule.

CW is then defined as the width in granules of the default fixed pitch printer font and this is used for microspacing.

If proportional printing is required the character widths must be determined. These may be available from the printer documentation, but more often this is not the case. A file, called 'WIDTHS.PP' is provided on one of the Protext discs to help in determining the widths. This file should be loaded into Protext and printed. This will print 60 of each character. Each line should be measured in inches and the result multiplied by the figure in the 'gpsi' column in the above table. The measurement should be accurate to 1/gpsi inches, e.g. for a laser printer it should be measured in 1/5th inch units. This gives the width of the character on that line in granules and is the figure to be entered in the printer driver using the WIDTH command.

Finally, the proportional character width (PW) must be defined. This is analagous to CW and is, in a sense, the 'average' width of a proportional font's characters in granules. This determines the width of the printed line.

When using a fixed pitch font the number of characters printed on a line is simply the width of the ruler in characters and the width is this number multiplied by CW. When using a proportional font the number of characters printed is variable, so a method is needed to relate the ruler to the printed width. This is PW, the width of the printed line is the width of the ruler line in characters multiplied by PW. To some extent, PW is chosen for convenience to reflect the average character width.

Using a new printer driver

The new printer driver may be loaded at any time with the PRINTER command. Alternatively, once the file has been tested to see it gives the required results, CONFIG may be used to make the new driver into the default driver to be loaded automatically when Protext is loaded.

31. Hints and Tips

This chapter covers certain problems that sometimes occur when using Protext and suggests the most common solutions.

Printer Problems

The printer doesn't print anything

Check the following:

Is the printer power lead plugged in?
Is the printer cable connected to both printer and computer?
Is the correct printer port on the computer being used?
Is the cable plugged in the right way up?
Is there paper in the printer?
Has the printer been switched offline?
Has a configuration file specifying a non-existent output port been loaded, or output been redirected to the wrong printer?

A blank line is printed after each line of text.

The printer is automatically feeding the paper an extra line each time. This is a feature of printers that is needed by some software, but must be disabled for Protext.

This feature is usually controlled by a switch ('DIP switch') in or on the printer. The printer manual will give details. The switch should be set to 'no automatic line feeds'.

Should this solution fail, there is a software means to prevent the extra line feeds. Use CONFIG, printer driver options, and set 'ignore line feeds' to 'YES'.

£ signs or accented letters not printed correctly

The wrong printer driver is being used.

If using a dot matrix printer note that many printers are capable of working in either Epson mode or IBM mode. This is normally selected by a switch on the printer. If the printer has Epson mode selected, then the EPSON printer driver (or similar) should be used. If IBM mode is selected the IBM printer driver (or similar) should be selected.

Italic print does not work

The printer may be set to IBM mode, which does not support italic print.

Line drawing mode - boxes do not print correctly

In order to print line graphics the printer must be one of: IBM compatible, Epson compatible, HP LaserJet compatible. Many dot matrix printers can be set to Epson or IBM mode - it is important that the printer driver matches the printer setting.

Serial printer not working

Serial printers give a lot more trouble than parallel printers, and parallel should always be chosen where there is a choice. Most problems with serial printers are caused by one of two reasons:

The printer cable may be incorrectly wired. Consult the dealer who supplied the equipment or a hardware specialist.

The serial port on the computer may be incorrectly configured. In particular the baud rate must be set to match that of the printer. This will often be 9600 baud, but the printer manual should be consulted.

Sheet feeder not feeding the pages through
Laser printer does not start printing for several seconds

These problems are usually solved by configuring Protext to send form feed characters after every page. This can be done from the 'layout options' menu in CONFIG.

The letter 'P' is printed at the start of each document

The wrong printer driver is being used.

If using a daisy wheel printer change to the STDDAISY printer driver.

Protext is cancelling the effect of control panel settings
Any other printer problems

There is an option in CONFIG (printer driver options) to cause Protext to ignore the reset code. This is a special code in each printer driver that is printed at the start of each document. If it often worthwhile trying this option when experiencing problems with printing. If this cures the problem it may be that the wrong printer driver was being used.

Single sheets, background printing, form feeds, paper sensing, PP

This section summarises for ease of reference the inter-relationship between these items. This may seem complicated but the various options have been put together to allow the greatest flexibility and to cope with many different types of printer.

With background printing on (the default)

Initially printing will be continuous (CP ON) with form feeds off (FF OFF) and paper sensing off (PS OFF).

Form feeds are off because the printer page length may not match the Protext page length, particularly when using continuous stationery. This is less critical when using single sheet paper because the form feed simply ejects the paper rather than feeding it a specified distance.

Form feeds will normally be turned on (FF ON) when using (a) a laser printer, (b) a cut sheet feeder. In these cases, though single sheets are being used it is usual to leave continuous printing on (CP ON) because the sheets are fed automatically.

Single sheet printing (CP OFF) will normally be selected when using manually fed single sheets of paper. This will automatically cause form feeds to be sent, so it is not necessary to select form feeds. A message will appear after each page is printed: "Press RETURN after inserting next sheet of paper, or ESC to stop printing".

The PP command is used when it is required to skip some pages. A message will appear before each page is put into the background printing buffer: "Page nn. Press SPACE to print, RETURN to skip, ESC to end". The message after each page is printed will still appear if single sheet printing has been selected.

If paper sensing is selected (PS ON) and the printer driver supports this, the first message (after each page) is disabled. Instead the printer is turned off line after each page is printed.

With background printing off

If single sheet printing is selected (CP OFF) the second message will appear before each page is printed, allowing pages to be skipped.

The PP command can be used to skip pages if continuous printing is on.

How to make your printer default to Quality print

Using a proportional daisy wheel printer

If your printer has a front panel or switch which allows selection of fonts or quality printing, this will usually be disregarded by Protext because Protext v5 needs to know which is the initial font.

Disabling the reset code (in CONFIG) will appear to solve this problem. However this method is not recommended as it will prevent Protext from correctly formatting the text and in some cases will also prevent some characters from printing.

The best solution to this is to add a line to the printer driver being used to set Quality as the default font. The line that should be added is:

```
FD Q
```

Similarly if you are using a daisy wheel printer and always use a proportional daisy wheel it is best to add the following line to the printer driver:

```
FD P
```

Keyboard Problems

The £ key produces

This probably means that the U.S. keyboard layout has been selected. Use CONFIG (keyboard options menu) to select the correct keyboard language.

Some keys produce the wrong characters such as \¦ # ~ *PC only*

PC	*The wrong keyboard type has been selected. Use CONFIG to select one of the other two types.*

Problems accessing memory resident programs *PC only*

PC *Although Protext is compatible with most 'pop-up' programs, there may be problems with some. To use such programs it can be useful to switch off the PROTEXT keyboard driver whilst using PROTEXT. A special command token (769) will do this. To use this the token must be put on a function key, for example:*

```
KEY s10 ^765^^769^
```

Edit mode must be selected to use the keyboard switch, and the token 765 ensures that this is the case. After returning to PROTEXT pressing the function key again will restore the PROTEXT driver. Alternatively, before pressing the 'hot key' type the following in command mode:

```
#pause
```

This will cause the resident program to use the MS-DOS keyboard.

Some ALT key symbols or functions do not work
How to remove existing macro definitions

The ALT key combinations are defined as macros, but many of them have default meanings in Protext (for example ALT-U for underline and ALT-F to change the number of lines displayed). If macros have been defined using these keys and saved in a key file the original functions will no longer work.

Some of the ALT key functions have been changed from earlier versions of Protext and so the old definitions may be stored in the key files. The easiest way to clear all the definitions is to type the commands:

```
CLRK
SVK        (press RETURN twice)
```

This will clear any user-defined macros, so it may be preferable to remove individual macro definitions. As an example the commands to restore ALT-U to the Protext function of underline are given:

```
KEY U      (press RETURN twice)
SVK        (press RETURN twice)
```

This saves the key file ready for next time Protext is used.

To reload the keys immediately use:

```
CLRK
LDK        (press RETURN twice)
```

The 'CRLK' restores the default value for ALT-U.

General Hints

Printing lines starting with '>'

A line starting with '>' will be taken as a stored command line. If it is required to print this line instead then insert a spell ignore marker at the start of the line by typing ALT-=.

Printing the whole text in condensed or elite, including header

If a control code is put on a line before the header stored command, the header will not appear until page 2.

The solution is to put the control code on the line after the header (the first text line), with NO OTHER characters on this line. Protext will then send the control code before the header.

For example, to print the text all in condensed, enter:

```
>HE    page %
c
```

Alternatively, the stored command '>FD' can be used:

```
>FD c
```

To print a list of the files on disc

Use the following three commands:

```
PRON
CAT
PROFF
```

A1. Summary of Protext Commands

This summary is divided into the following categories and gives concise details of the syntax of all commands used by Protext.

(a) Edit mode commands
(b) Stored commands
(c) Command mode commands
(d) External utility program commands

(a) Edit mode

The commands marked '*' are not available in program mode.

Altern.	Key		Function
Cursor movement			
	←		Move left one character
	→		Move right one character
	↑		Move up one character
	↓		Move down one character
	SHIFT-←		Move left one word
	SHIFT-→		Move right one word
	SHIFT-↑		Scroll back 1 line
	SHIFT-↓		Scroll forward 1 line
HOME	CTRL-←		Move to start of line
END	CTRL-→		Move to end of line
PgUp	CTRL-↑		Scroll back one screenful
PgDn	CTRL-↓		Scroll forward one screenful
	SHIFT-CTRL-←		Move to start of sentence
	SHIFT-CTRL-→		Move to end of sentence
	CTRL-[		Move to start of memory
	CTRL-]		Move to end of memory
CTRL-HOME	CTRL-[CTRL-[		Move to start of document
CTRL-END	CTRL-] CTRL-]		Move to end of document
	CTRL-(	*	Move back one page
	CTRL-)	*	Move forward one page
	SHIFT-CTRL-(	*	Move back to start of column
	SHIFT-CTRL-)	*	Move to start of next column
	CTRL-<		Move back one paragraph
	CTRL->		Move forward one paragraph

	CTRL-L	Go to last position
	CTRL-G	Go to line/page/column
ENTER	RETURN	Insert mode - split line and move to left margin of next line Overwrite mode - move to left margin of next line
	SHIFT-RETURN	Move to left margin of next line
	CTRL-RETURN	Move to left margin of next line
	TAB	Insert mode - insert tab character Overwrite mode - move to next tab
	SHIFT-TAB	Insert mode - move to next tab Overwrite mode - insert tab character
	ALT-TAB	Insert indent tab

Insertion and deletion

	CTRL-I	Insert line
CTRL-Kpd*	CTRL-f3	Delete line
	CTRL-←DEL	Delete to start of line
CTRL-DEL→	CTRL-E	Delete to end of line
	SHIFT-CTRL-←DEL	Delete to start of sentence
	SHIFT-CTRL-E	Delete to end of sentence
Backspace	←DEL	Delete character before cursor
DEL→	DEL	Delete character at cursor
	SHIFT-←DEL	Delete word left
SHIFT-DEL→	SHIFT-DEL	Delete word right
INS	CTRL-TAB	Toggle insert/overwrite mode
	CTRL-A	Transpose (Alternate) characters
	CTRL-U	Undo last delete operation
	CTRL-*	Split line at cursor
	CTRL-+	Join lines

Block commands

f9	CTRL-Z	Set or clear block markers
CTRL-f9	CTRL-K	Clear all block markers
CTRL-f10	CTRL-M	Move block
CTRL-INS	f10	Copy block
CTRL-f4	CTRL-DEL	Delete block
	CTRL-O	Copy block from other document

Formatting and rulers

	CTRL-F	*	Format to end of paragraph
	SHIFT-CTRL-F	*	Format whole paragraph
	CTRL-C	*	Centre line
	CTRL-D	*	Copy default ruler (first active ruler)
	CTRL-R	*	Copy previous ruler but one
	CTRL-J	*	Justify on/off
	CTRL-W	*	Word wrap on/off
	SHIFT-CTRL-R	*	Auto-reformatting on/off
	SHIFT-CTRL-I		Auto indent on/off (program mode)

Find and replace, marker

Keypad+	f6		Next find
Keypad-	f5		Previous find
	SHIFT-CTRL-f6		Find word at cursor
	SHIFT-CTRL-f5		Find word at cursor backwards
f8 n	CTRL-@ n		Set/Go to marker (0 to 9)
f8 ?	CTRL-@ ?		Insert multiple marker
f8 [	CTRL-@ [		Go to [block marker
f8]	CTRL-@]		Go to] block marker
f8 L	CTRL-@ L	*	Go to left margin
f8 R	CTRL-@ R	*	Go to right margin
f8 F	CTRL-@ F	*	Set footnote marker
CTRL-Kpd+	CTRL-f6		Go to next marker
CTRL-Kpd-	CTRL-f5		Go to previous marker

Other commands

	CTRL-N space	*	Non-break space
	CTRL-N hyphen	*	Non-break hyphen
	CTRL-P	*	Page mode on/off
	CTRL-V P	*	Page breaks display on/off
	CTRL-V R	*	Ruler line visible/hidden
	CTRL-V S		Hard spaces visible/hidden
	CTRL-V T		Tabs and returns visible/hidden
f4	CTRL-V V		Control codes visible/hidden
	CTRL-T		Reserved for future expansion
f7	CTRL-X	*	Insert printer control code
f7 B	ALT-B	*	Insert bold code
f7 I	ALT-I	*	Insert italic code

f7 U	ALT-U	*	Insert underline code
f7 =	ALT-=	*	Insert spell ignore marker
f7 *	ALT-W	*	Insert index word marker
f7 #	ALT-P	*	Insert index phrase marker
	CTRL-Y		Switch between documents in memory
	SHIFT-CTRL-Y		Exchange top and bottom files
	SHIFT-CTRL-X		Split/unsplit screen
	CTRL-/		Convert letter to upper case
	CTRL-\		Convert letter to lower case
	SHIFT-CTRL-/		Convert word to upper case
	SHIFT-CTRL-\		Convert word to lower case
	CTRL-hyphen	*	Insert soft hyphen
	CTRL-space		Insert a space
SH-CTRL-M	CTRL-f1		Macro record mode on/off
	ALT-letter		Invoke macro
	f3		*Use menus (not Amiga)*
	Scroll Lock		*Page mode on/off (PC)*
	CTRL-HELP		*Page mode on/off (ST / Amiga)*
	ESC		Enter command mode

Spell checking

CTRL-Q		Spell check single word
CTRL-S	*	Spell check from cursor to end of file
SHIFT-CTRL-S	*	Spell check whilst typing on/off

Extra characters

ALT-H	½	Half
ALT-Q	¼	Quarter
ALT-<	«	Open quotes (French)
ALT->	»	Close quotes (French)
ALT-!	¡	Inverted exclamation mark
ALT-?	¿	Inverted question mark
ALT-↑	↑	Up arrow
ALT-↓	↓	Down arrow
ALT-←	←	Left arrow
ALT-→	→	Right arrow

	SHIFT-ALT-Keypad numbers	*Single line graphics (PC)*
	SHIFT-CTRL-Keypad numbers	*Single line graphics (PC)*
SH-CTRL-L	CTRL-f7	Line drawing mode on/off
SH-CTRL-A	CTRL-f8 char	Line drawing using character
	ALT-cursor keys	Draw line
	SHIFT-ALT-fn	Set keyboard to language n

Accents and special characters

ALT-'	´	acute
SHIFT-ALT-~	˘	breve
SHIFT-ALT-^	ˇ	caron (háček)
ALT-^	^	circumflex
ALT-"	¨	diaeresis/umlaut
SHIFT-ALT-'	˝	double acute
ALT-\	`	grave
ALT-_	_	macron
SHIFT-ALT-.	˙	overdot
ALT-%	°	ring
ALT-~	~	tilde
SHIFT-ALT-,	¸	cedilla
ALT-;	˛	ogonek
(SHIFT-) ALT-C	ç	c cedilla
(SHIFT-) ALT-E	æ	ae diphthong
(SHIFT-) ALT-L	ł	l slash (Polish)
(SHIFT-) ALT-N	ñ	n tilde
(SHIFT-) ALT-O	ø	o slash
ALT-S	ß	double s (German)

Miscellany

ALT-D	insert date into text
ALT-T	insert time into text
ALT-F	*25 and 43/50 line toggle (PC EGA/VGA, ST mono)*
ALT-G	*graphics/text mode (PC)*

Note: Accents on their own may be produced by typing a space and then keying the accent keys.

Note: Those commands with '(SHIFT-)' in brackets produce upper case characters when SHIFT is pressed as well.

(b) Stored commands

Key:	a	an ASCII character
	f	a filename
	n	an integer between 0 and 255
	v	a variable identifier
	(x)	an optional parameter
	{x}	a parameter that may occur 0 or more times
	expr	a string expression
	text	a string of chars, optionally in quotes
	cond	a conditional string expression

Where more than one parameter is listed on a command line, the parameters may be separated by spaces, commas, or equals signs. This is not shown in the syntax descriptions below.

Stored commands are broken down into a number of categories according to their most common usage. Within each section they are arranged alphabetically. A separate appendix gives full technical details of all the stored commands in strictly alphabetical order. The subheadings are:-

Paper layout commands
Page formatting commands
Messaging commands
Miscellaneous general purpose
Printer control commands
Variable and data input - mail merging
Conditional printing and mail merging

The commands marked '#' take effect at the next new page printed.

Paper layout commands

# >BM n	Bottom margin.
>EM n	Even side margin.
# >FM n	Footer margin.
# >HM n	Header margin.
>OM n	Odd side margin.
# >PL n	Page length.
>SM n	Side margin.
# >TM n	Top margin.
# >ZM	Zero margins.

Note: all parameters are in lines or columns as appropriate.

Page formatting commands

>CE text	Centre line.
# >CP ON/OFF	Continuous/Single sheet printing.
>DE	Dump endnotes at foot of page.
>EA nn	End at page no.
>EC	End column.
# >EF text	Define even footer text and turn footers on.
# >EH text	Define even header text and turn headers on.
>EN ON/OFF	Turn use of endnotes instead of footnotes on/off.
>EP (n)	Even page throw (can be conditional).
>FE	End footnote definition.
# >FF ON/OFF	Form feeds enabled/disabled.
>FN	Start multi-line footnote definition.
>FN text	Define single line footnote.
# >FO text	Define footer text and turn footers on.
# >FO ON/OFF	Turn footers on/off.
>FP ON/OFF	Formatting whilst printing on/off.
>FX ON/OFF	Fix text (will not be reformatted during reformatting).
# >HE text	Define header text and turn headers on.
# >HE ON/OFF	Turn headers on/off.
>IP ON/OFF	Inter-paragraph space increased by half a line on/off.
>LS n	Line spacing.
# >LF ON/OFF	Turns last page footer on/off.
>NC n	Number of copies.
>NP ON/OFF	Enable/disable new page at end of printing.
# >OF text	Define odd footer text and turn footers on.
# >OH text	Define odd header text and turn headers on.

>OP (n)	Odd page throw (can be conditional).
>PA (n)	Page throw (can be conditional).
>PC n (n)	Print columns.
# >PE ON/OFF	Print even pages only.
# >PN nn	Page number of next page.
# >PO ON/OFF	Print odd pages only.
>PS ON/OFF	Paper sensing on/off.
>RJ ON/OFF	Right justifying on/off.
>SA nn	Start at page no.

Miscellaneous commands

>CO text	Comment line.
>CS text	Clear screen and display message.
>DM text	Display message.
>EX (?)text	Execute command.
>GC f <opts>	Generate contents.
>GI f	Generate index.
>IB (?)f	Insert binary file and copy directly to output.
>IN f	Insert file.
>SC ON/OFF	Spell checking on/off.
>SL n {f}	Define spell language and dictionaries.
>ST text	Stop printing and display text.
>TL a	Define tab leader character.
>WC	Write file close.
>WF f (A)	Open file for writing (appending).
>WF ON/OFF	Writing to file on/off.
>WM text	Write message to file - used with WF.
>WT text	Wait and display text.

Printer control commands

>CC a {n} (;a{n})	Define printer control code.
>CW n	Define microspace character width.
>FD {a}	Define default font.
>MC n {n}	Define microspace code sequence.
>MF n	Define microspace factor.
>MM n	Set microspacing method.
>MP ON/OFF	Multiple copy printing on/off.
>MS ON/OFF	Microspacing on/off.
>OC n {n}	Output codes to printer.
>PP ON/OFF	Proportional spacing on/off.

>PR f	Load printer driver.
>PW n	Proportional character width.
>RC a {n}	Redefine character.

Variable and data input - mail merging

>AV v {v}	Ask for variables.
>CF	Close data file.
>DC ,¦.	Set the decimal character.
>DF f {f}	Define data file.
>DP n¦R	Set number of decimal places or Roman numerals.
>DU	Dumps currently defined variables to screen.
>RU v {v}	Read variables unconditionally.
>RV v {v}	Read variables and pad with nulls.
>SV v=expr	Set variables.

Conditional printing and mail merging

>EI	End of ID, IE, IF or IU block.
>EL	Else - print block if previous IF condition false.
>ID v	Print block if variable defined.
>IE	Print block if data file exhausted.
>IF cond	Print block if condition true.
>IU v	Print block if variable undefined.
>RP	Repeat - until following UN condition is true.
>SK cond	Skip printing if condition true.
>UN cond	Until condition is true - repeat from RP command.

Note: If column one contains '>', and columns two and three each contain one of: space, 'L', '-', or '!' then the line is a ruler line.

(c) Command Mode

The following commands are listed according to a number of categories and alphabetically within the category for ease of location.

Key to parameters:

a	an ASCII character
dr	drive letter
f	a filename (may include drive and/or directory path)
af	ambiguous filename (may contain wildcards)
cfgfile	configuration file
dict	dictionary filename
inf	input filename
outf	output filename
newf	new filename
oldf	old filename
outdict	resultant dictionary
n	an integer between 0 and 255
nn	an integer between 0 and 65535
wordpat	a sequence of characters

Items marked with '<..>' are mandatory. Those marked with '(..)' are optional.

Command		Abbr.	Description
f3			*PC/ST/Archimedes: call Main Menu.*
PM			Call Print Menu.

Document handling

CLEAR (ALL)		-	Clear current text (or all files).
DOC		-	Select document editing mode.
EXT		-	Set filename extension for load/merge.
LINK	ON ¦ OFF	-	Link two files for scrolling.
LOAD	<f>	L	Load new file.
MERGE	<f>	M	Merge file with current text.
NAME	<f>	N	Assign name to current file.
PRINTF	<outf> (inf)	PF	Print to disc, including printer control codes, opt. from disc.

PRINTFB	<outf>	PFB	Print block to disc, including printer control codes.
PROG		-	Select program editing mode.
SAVEA	<outf>	SA	Save to disc as ASCII file.
SAVEAB	<outf>	SAB	Save block to disc as ASCII file.
SAVE	<f>	S	Save text.
SAVEB	<f>	SB	Save block.
SHOW		SH	Show list of open files.
SPLIT	<f> <nn>	SPL	Split large file.
SPOOL	<f>	SPON	Echo all screen output to a file.
SPOOLOFF		SPOFF	Turn off echo to file.
SWAP	<n>¦<f>	SW	Swap between two documents in memory.
TYPE	<f>	T	Types the file to screen.
WINDOW	<n>	WIN	Set size of split screen window.

Printing options

PRINT	(f)	P	Print file, optionally from disc.
PRINTB		PB	Print marked block of text only.
PRINTF	<outf> (inf)	PF	Print to disc, including printer control codes. Optionally from disc.
PRINTFB	<outf>	PFB	Print block to disc, including printer control codes.
PRINTP	(f) (page list)	PP	Print pages selectively, opt. from disc.
PRINTPQ	(f) (page list)	PPQ	Print pages in NLQ, opt. from disc.
PRINTPS	(f) (page list)	PPS	Print pages to screen, opt. from disc.
PRINTQ	(f)	PQ	Print, NLQ mode, opt. from disc.
PRINTQB		PQB	Print block, NLQ.
PRINTS	(f)	PS	Print text to screen, opt. from disc.
PRINTSB		PSB	Print block to screen.

Printer control commands

ABANDON		AB	Abandon printing and clear buffer.
BACK	ON ¦ OFF	-	Turn background printing on/off.
CONT		CO	Continue printing from buffer.
PARALLEL	*(n)*	*PAR*	*PC/ST: Select parallel (Centronics) port.*
PRINTER	<f>	PR	Load printer driver.
SERIAL	*(n)*	*SER*	*PC/ST: Select serial port.*
STOP		ST	Stop background printing.

Sundry printing options

PRINTON	PRON	Echo all screen output to printer.
PRINTOFF	PROFF	Turn off echo to printer.
TYPEWRITER (n)	TW	Typewriter mode.

Text manipulation and formatting

FIND	F	Find string.
FINDB	FB	Find string within block.
FIX	-	Remove tabs, soft spaces and soft returns.
FIXB	-	As for FIX, but on marked block.
FORMAT	FT	Format text.
FORMATB	FTB	Format block.
NUMBER	NUM	Number lines of text.
NUMBERB	NUMB	Number lines of text in marked block only.
REPLACE	R	Find and replace string.
REPLACEB	RB	Find and replace string within block.
TAB	-	Set tabs for program mode.

Spell checking

ANAGRAM	<wordpat>(dict)	AN	Search for words containing same characters.
BUILD	<f> <outf>	BU	Build file of unrecognised words.
COUNTD	(dict)	COD	Count words in dictionary.
DELWORDS	<f>(dict)	DW	Delete words in text file from dictionary.
FINDW	<wordpat>(dict)	FW	Search for words with same character pattern.
INSWORDS	<f>(dict)	IW	Add words in text file to dictionary.
LISTD	(dict)(word)	LD	List dictionary.
LOOKUP	<word>(dict)	LK	Search for similar words.
MAKED	<f> <outdict>	MKD	Make dictionary from text file.
SPELL	<f>	SC	Two pass spell check of file.
SPELLB		SCB	Spell check current block.

Macro and Exec commands

CLEARKEYS	(N)	CLRK	Clear keys. Set defaults or none.
CODE		-	Display token returned by a key.
ECHO	ON ¦ OFF	-	Turn command echo on or off.
EXEC	<f>	X	Execute file of commands.
EXECS	<string>	XS	Execute string of commands.
LMACROS		LM	List defined macros.
LOADKEYS	<f>	LDK	Load key definitions.
MACRO	(a)<a or n> <macro>	KEY	Define macro.
SAVEKEYS	<f>	SVK	Save key definitions.

Miscellany

CALC	<expression>	CA	Calculator.
CLS		-	Clear screen.
COUNT		CT	Count words.
COUNTB		CTB	Count words in block.
DATE	*<date>*	-	*ST: set date*
FF		-	Send a form feed to the printer.
GOTO	<a> <nn>	G	Goto Page/Line/Column number.
HELP	(topic)	H	Display HELP on given topic.
INK	*<ink> <colour>*	-	*ST: assign colour to ink.*
KEYB	<n>	-	Select keyboard type.
KEYDEF	<n> <codes>	-	Redefine a key.
LANG	<n>	-	Select keyboard language.
LOADCFG	<cfgfile>	LCFG	Load Configuration (CFG) file.
PAUSE		-	Pause (optionally with message).
QUIT		Q	Quit Protext.
STATUS		STAT	Lists useful configuration settings.
SYMBOL	<n> <codes>	SYM	Redefine a character on screen.
TIME	*<time>*	-	*ST: set time*

Disc management commands

A, B, C, D, or E			Drives A to E may be selected by just typing the letter.
ACCESS	(af)	ACC	Set file(s) to read-write status.
CAT	(af)	-	Catalogue directory.
CHDIR	< path >	CD	Change current directory.
COPY	*<af> (<path> ¦ <f>)*	-	*ST/Amiga: Copy files.*
DIR	(af)	-	Display directory listing.
DRIVE	(dr)	DR	Select specified drive.
ERASE	< af >	DEL	Delete a file or files.
MKDIR	< path >	MD	Create a new directory.
PROTECT	(af)	PROT	Set file(s) to read-only status.
RENAME	< oldf > < newf >	REN	Rename a file.
RMDIR	< path >	RD	Remove a directory.
TOUCH	*<af>*	-	*ST: Set file time to current date and time.*
UPDATE	*<af> (<path> ¦ <f>)*	*UP*	*ST/Amiga: Update changed files.*
0		-	*Amiga/Archimedes: Select drive 0*
1		-	*Amiga/Archimedes: Select drive 1*
2		-	*Amiga/Archimedes: Select drive 2*
3-7		-	*Archimedes: Select drive 3-7*

(d) External utility program commands

The following commands call programs which must be available to Protext. In other words the specified program files must be available on disc.

CONFIG	CFG	Menu driven configuration program.
CONVERT	CV	File conversion program.
FSORT	-	File sorting program.

A2. Protext - Compatibility between versions

Text files created with any version of Protext may be edited using any other version. The various versions of Protext will be referred to as follows. This section highlights differences between version 5 and earlier versions that may be relevant when transferring files between versions. See also 'File Conversion'.

version 1	Amstrad CPC (Amsdos) version (1985)
version 2	Amstrad CP/M+ (PCW & CPC6128) (1986)
version 3	MS-DOS (1987)
version 3.5	Atari ST (1988)
version 4.0	MS-DOS, Atari ST (1988)
version 4.1	Amiga (1988)
version 4.2	MS-DOS, Atari ST, Amiga (1989)
version 4.3	MS-DOS, Atari ST, Amiga (1990)
version 5.0	MS-DOS, Atari ST, Amiga, Archimedes (1990)

1. Due to the different disc formats of the various computers, it is not always possible to read or write files directly onto compatible discs. However the ST, Amiga and Archimedes will read 3½" PC discs. In other cases files may be transferred via the RS232 Serial ports with suitable software.

2. Version 4 and above support some additional stored commands. If the document is required to be printed using the other versions of Protext these commands should not be used. Note that with version 1 the AMSDOS program PROMERGE or PROMERGE PLUS will be required. The stored commands available from version 2 only: IE, PE, PO, PP, WF, WC, WM. In addition the following are available from version 3 only: DC, DP, DU, LF, MF, MM, PW. The following are available from versions 3.5 only: FN, multi-line headers and footers. Centre tabs in ruler lines are from 4.3 only.

3. Version 1 uses language stored commands (DA, EN, FR, GE, IT, SP, SW). In all later versions all characters can be printed without commands to select the printer font and these commands must be removed.

4. Certain special characters supported by version 4 are not recognised by some earlier versions, although they will not be corrupted if a file containing them is loaded into other versions of Protext. Version 3.5 and above: footnote markers. Version 3 and above: spell ignore markers, index markers. Version 2 and above: All characters with accents, the special characters available on function keys, the arrow characters, non-break hyphens, multiple markers.

5. Version 5 and above support additional stored commands (DE, EC, EN, FE, GC, GI, IB, IP, MP, PC, SC, SL, TL), extensions to FN, DP and the adding of accents to any character.

A3. Alternative Keyboard Commands

Using the MS-DOS keyboard driver *(PC only)*

Protext normally uses a specially written keyboard driver. It is possible, however, to dispense with this and use the standard MS-DOS keyboard driver. There are two possible reasons why this may be required:

(a) The Protext keyboard driver does not work. This may occur on certain computers which are not fully compatible with the IBM PC keyboard.

(b) Protext is being used in conjunction with other software which uses the standard keyboard driver. This is not a problem in itself, but some of the keys used may be similar if the Protext keyboard driver is turned off.

To disable the Protext keyboard driver use the CONFIG program and change the option 'Protext keyboard driver' to 'OFF'. The differences in the keys used are summarised below:

(i) ALT <letter> is used in place of CTRL <letter>. Other commands transferred to ALT are: ALT-<hyphen> to insert a soft hyphen, ALT-(and ALT-) to move by a page, ALT-+ to join lines, ALT- to split a line. CTRL-←DEL and CTRL-RETURN (but not SHIFT-RETURN) have the same function with either keyboard driver. SHIFT-TAB does not operate.*

(ii) The numeric pad keys together with SHIFT give numbers. Keypad + and keypad- give + and -. Home, End, PgUp, PgDn, Ins and Del work in the same way as the Protext driver. Other keypad functions are: CTRL-← and CTRL-→ for word left and right. The arrow characters are not available with ALT, instead they may be inserted by ALT-V <arrow>.

(iii) A number of functions and special characters are available on function keys. The function keys with CTRL work as with the Protext keyboard driver.

	normal	*alt*	*shift*
f1	*help*	*upper case*	*æ*
f2	*catalogue files*	*lower case*	*Æ*
f3	*enter menu*	*delete line*	*ç*
f4	*view codes on/off*	*delete block*	*Ç*
f5	*previous find*	*previous marker*	*ß*
f6	*next find*	*next marker*	*¿*
f7	*printer code*	*back paragraph*	*«*
f8	*set/goto marker*	*forward paragraph*	*»*
f9	*set block marker*	*delete word right*	*½*
f10	*copy block*	*delete word left*	*¶*

(iv) A few commands are not assigned to keys and these should be used from the menus. Accents should also be obtained from the menus.

WordStar emulation

The commands listed below are the only differences from normal Protext operation. The ALT key is still used for macros. Commands not available from the keyboard in WordStar emulation should be used from the menus.

The following WordStar commands are supported:

CTRL-A	move back a word
CTRL-B	format paragraph
CTRL-C	move forward a screen
CTRL-D	cursor right
CTRL-E	cursor up
CTRL-F	move forward a word
CTRL-G	delete forwards
CTRL-H	delete backwards
CTRL-I	TAB
CTRL-L	next find
CTRL-N	split line
CTRL-P	printer control code
CTRL-R	move back a screen
CTRL-S	cursor left
CTRL-T	delete word to right
CTRL-U	undelete
CTRL-V	insert/overwrite
CTRL-W	scroll up 1 line
CTRL-X	cursor down
CTRL-Y	delete line
CTRL-Z	scroll down 1 line

CTRL-Q C	move to end of text
CTRL-Q D	move to end of line
CTRL-Q E	move to top of screen
CTRL-Q P	move to last position
CTRL-Q R	move to start of text
CTRL-Q S	move to start of line
CTRL-Q V	move to last position
CTRL-Q X	move to bottom of screen
CTRL-Q Y	delete to end of line
CTRL-Q DEL	delete to start of line
CTRL-O C	centre line
CTRL-O J	justify on/off
CTRL-O T	ruler on/off
CTRL-O W	wordwrap on/off
CTRL-K B	set marker
CTRL-K C	copy block
CTRL-K N	box mode on/off
CTRL-K K	set marker
CTRL-K V	move block
CTRL-K Y	delete block

A4. Sample Merging Documents

This chapter consists of a number of sample merging documents intended to give some idea of the variety of possible uses for Protext. Some of them are complete documents in their own right, but many are working extracts from larger documents, intended to demonstrate solutions to a variety of problems. The first comment line in each program describes the use of the document and the next line describes the type of records required in the data file. There is a brief description of how the program works and what it does, after each document.

The last example is worthy of study, not so much for its usefulness, but as an example of methods of programming and the versatility of Protext.

The first example is only for reference and is 'EXAMPLE6' from the earlier chapters, listed in full.

```
>CO    EXAMPLE6
>CO data file requires name, tel, 4 address lines
>PL 24       ;for screen display
>CP OFF      ;purposes only
>DF datfile3 ;open data file
>RU name tel addr1 addr2 addr3 addr4 dummy ;read data fields
>SK addr4 > ""           ;check for last address line empty. If so, skip
>AV "Enter Date " date   ;request date
>SV year="1986"          ;set the year

&name&
&addr1&
&addr2&
&addr3&
&addr4&

Dear &name&,
>CE  Renewal date - &date& &year&
    Thank you for your letter about the insurance for &addr1& &addr2& &addr3&
&addr4&. We think that you will find the rates quoted......

>IF tel>""          ; if telephone - print
    If you would be kind enough to telephone us......
>EL                 ; otherwise - print
    If you would be kind enough to write to us ......
>EI                 ;end of IF EL block
```

```
>CO  Invoice printer and calculator
>CO  No data file required
>SV gnet=0  gvat=0  gtotinc=0
>SV vatrate=0.15
Description  Retail Quant   Disc  Total     Net          VAT     TOTAL
>RP
>AV "Description of goods "descrip 12
>AV "Retail price "ret "Quantity "quant "Discount "disc
>IF disc=""
>SV disc=0
>EI
>SV totret=ret*quant off=totret/100*disc net=totret-off vat=net*vatrate
>SV totinc=net+vat gnet=gnet+net gvat=gvat+vat gtotinc=gtotinc+totinc
>----------------.-----.-------.------.--------.----------.--------.-R
&descrip&   &ret& &quant& &disc& &totret& &net&       &vat& &totinc&
>AV "Another entry? Y/N " yeno
>UN yeno[1]<>"Y"
                                        ------------------------------
                                TOTALS   &gnet&     &gvat& &gtotinc&
                                        ------------------------------
>ST
```

The above template file creates invoices. Based on information requested, it will calculate the various totals, net values and VAT. After each item the option of entering another item is offered. Any answer but 'Y' will stop further entry and the grand totals will be printed. Use the PRINT command.

```
>>>     This example is part of an invoice generating template
>>> no data file is required
>RP      ; repeat enter order until correct
>AV "Protext order (PC, Atari, Amiga): ",orderp
>SV orderp=orderp+" 0 0 0"  ; default quantities of zero
>IF orderp[1]=" "           ; if no figures were entered
>SV orderp=orderp[2:]       ; strip leading space
>EI
>SV ep=orderp[w1] dp=orderp[w2] cp=orderp[w3]
>AV "Re-enter order (y/n)? ",again?
>UN again?[1]<>"Y"
```

This is an extract from another type of invoice generator. In the full template, the prices would be calculated from the quantities and the procedure repeated for other products. The prices would be stored in the template file as variables.

The quantities of each of the 3 formats of Protext are entered as a single string, e.g. '10 20 10', and this string is split into the 3 individual numbers. Each quantity defaults to zero if omitted. The whole is enclosed in a repeat loop to allow the order to be re-entered if a mistake was made.

```
>CO Data file creator
>CO No data file required.
>ZM                       ;set margins to zero
>CS Type END to stop
>AV "Enter name " name    ;ask for name
>IF name="end"            ;see if it equals end
>ST                       ;and if so stop
>EI
>AV "Telephone number " tel "Address 1 " addr1 "Address 2 " addr2
>AV "Address 3 " addr3 "Address 4 " addr4
>SV dummy=""
>IF '"' IN addr1          ;check for double quotes in first address
>SV addr1="'"+addr1+"'"   ;if so, wrap address in single quotes
>EL                       ;or
>SV addr1='"'+addr1+'"'   ;wrap address in double quotes
>EI
"!name!"
"!tel!"
!addr1!
"!addr2!"
"!addr3!"
"!addr4!"
!dummy!
>IN datamake              ;call file again.
```

The above file will create a data file of names, telephone numbers and addresses. The file must be saved on disc with the same name as used in the '>IN' command at the end. Typing 'end' in answer to 'Enter name' will close file and stop further processing. The 'IF' command checks for house names with double quotes and if found, wraps the data in single quotes. The remainder of the variables are wrapped in quotes at the printing stage. The last line calls the file from disc again. The file is created by using SAVEA with a filename.

```
>CO routine to separate off initials
>CO no data file required
>AV name             ;Request name to separate initials from.
>SV n1=name[w1] n2=name[w2] n3=name[w3] n4=name[w4]  ;split name
>IF n4=name[w-1]     ;check for same as last word in 'name'
>SV inits=n1+n2+n3       ;if so, 'inits' = n1, n2 and n3
>EI
>IF n3=name[w-1]     ;repeat for third word
>SV inits=n1+n2          ;if so, 'inits' = n1 and n2
>EI
>IF n2=name[w-1]     ;repeat for second word
>SV inits=n1             ;if so, 'inits' = n1
>EI
&inits&   &name&
```

This file is an example of manipulating variables, in this case to extract the initials in a person's name. It takes account of likely variations in the number of initials.

```
>CO Example routine to select one of four data files.
>CO 4 data files, called 'address.a-d', 'address.e-k', etc
>AV name
>SV suff="junk s-z l-r e-k a-d"
>RP
>SV suff=suff[w2:]
>UN name[1] >=suff[1]
>SV df="address."+suff[w1]
>DF &df&
```

This file makes use of a RP UN loop to repeatedly shorten a variable until the first letter of 'name' is greater than or equal to the first letter of the variable. 'junk' is simply there to be discarded on the first loop. Finally, joining of variables and text is used, followed by making use of the ability to use a variable in a command, to define the data file. This would be a suitable routine to select a data file to be read, according to the surname, enabling data to be kept in a number of files.

```
>CO using stored commands to set other stored commands.
>CO no data file
>AV "Side margin " sm
>AV "Page length " pl
>AV "Single sheet Y/N" cp
>AV "No. of copies " nc
>IF "Y" IN cp
>CP OFF
>EL
>CP ON
>EI
>SM &sm&
>PL &pl&
>NC &nc&
```

This one simply asks for a number of details about the options required and then uses the variables either as part of the stored command, or to determine the setting of the appropriate stored commands.

```
>CO EXAMPLE to reconstruct a data file for import to another database
>CO data file called original.dat is required
>ZM                          ; zero margins
>PL 7                        ; page length=No. of field lines
>DF original.dat             ; old data file
>RU initials,surname,add1 add2,add3,tel,dummy    ; read existing data
>CO  New file rearranges order, combines initials and surname and adds
>CO  a fourth address field
!surname! !initials!
!tel!
!add1!
!add2!
!add3!
!dummy!
!dummy!
```

The file is created by using the SAVEA command and specifying the new filename. This is a simple example only and any data may be manipulated as necessary. For example, dates may be converted from one format to another.

Note that the optional '!' variable markers are used to ensure that even empty fields are printed to the new file.

```
>>> Example to convert dates to suitable form for conditional comparison.
>>> or different format. No data file required
>>> Date form  01 Mar 86
>AV "Enter Date (dd mmm yy) " date
>IF date[2]=" "
>SV date="0"+date
>EI
>SV mon=date[w2] cal="Junk Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec"
>SV ncal="Junk 01 02 03 04 05 06 07 08 09 10 11 12"
>RP
>SV cal=cal[w2:] ncal=ncal[w2:]
>UN cal[w1]=mon
>SV ndate=date[w3]+ncal[w1]+date[w1]+" " ndate=ndate[1:6]
>SV mdate=date[w1]+":"+ncal[w1]+":"+date[w3]
Original - &date&   New form - &mdate&   Comparison form &ndate&
```

```
>CO                         Prime Number Generator
>DF primes ; never accessed so any filename on disc can be used
>ZM
>IU i
>SV i=3.00
>EL
>SV i=i+2
>EI
>SV j=3.00
>>>
>RP
>SV t=i/j
>SK t[w-1] = 0   ; if remainder zero
>IF j*j > i
>SV iint=i[w1]
&iint&
>SK 1=1
>EL
>SV j=j+2
>EI
>UN 0=1
```

This file calculates prime numbers and should be started with PS. The only point to note is the 'DF' command, which must have the name of a file on disc, even though it is not accessed. It is necessary, as it permits 'SK' to work.

A5. Simple letter example

The object of this chapter is to provide a guide to the typical procedures that would be followed when writing a document and printing it out.

We shall assume, for the purposes of this example, that Protext has not been used before and that you will not, therefore, have created any 'Template files' to store your most frequently used layouts. The default settings provided by Protext are ideal for creating a letter on single sheet A4 sized paper and 11 inch continuous stationery, so for this example, we shall assume that the letter is to be printed on A5 single sheet stationery.

As we shall use A5 stationery for part of the time, it is a good idea to start by creating a 'template file' to hold the required setting for page length, margins etc. before we continue with writing the letter. Having done this, whenever we want to write a letter on A5 stationery, it will only be necessary to load the correct template file at the start. Template files are ideal for storing the settings for any frequently used document layouts. If we ALWAYS used A5 single sheet paper, it would be more sensible to reconfigure the default settings with CONFIG, so that they were always present whenever Protext was loaded.

Creating a Template File

The object of the template file is to create, through the use of stored commands, the settings we require for page length, all the various top, bottom and side margins, suitable rulers and any other features which we might like to add, such as proportional printing, or double line spacing.

The first thing we need to know is the size of the paper we are going to print on. In the case of A5 paper, it is approximately 5.75" wide by just over 8" tall.

We should start by deciding how many characters across the paper we wish to print. We shall print our letter in the default Pica print style, which gives 10 characters per inch, which means that if we were to print from one edge of the paper to the other, that we would be able to print about 56 characters, but we obviously want a clear margin on either side. The default side margin setting in Protext is 5 characters (that is, 5 spaces will be printed from the left hand edge of the paper before the text is printed), which on a small sheet of paper is a bit large. Therefore we shall reduce the side margin to 4 with the SM command. As we also want a similarly sized margin on the right of the page, that means that we can only print 48 characters on a line (56-4-4), so we will want a ruler that is 48 columns wide. We also want to inset the start of each paragraph by 5 columns, so when we create our ruler, we shall insert a tab marker in column 5.

The next thing to consider is how many lines we can print on a page. The standard setting for line pitch (the number of lines printed per inch) is 6, so we should set the page length to 48 (8 x 6 lines per inch), but one thing we must also take into consideration is that most printers require a certain amount of the paper fed beyond the print head, in order that the bail bar can hold it. Our printer requires about 1", so the effective length of our page must be reduced by 6 lines to allow for this. We shall therefore add 6 to the bottom margin with the BM stored command, leaving the page length as 48 (the physical paper size).

The next items to consider are the top and bottom margins. As we are not bothered about using headers and footers, we can ignore the default settings for header and footer margins, as Protext will ignore them if they are not specifically turned on by defining them. The default settings for top and bottom margins are 3, which are ideal for use with continuous stationery, but we don't really require a top margin as we are going to position the paper at the right position, so we shall set it to zero with the TM command.

We must also consider the fact that it is not normally possible to print on the last few lines of a sheet of paper, as by that time the end will have fed past the roller and will no longer advance as it should. We could have allowed for this by reducing the number of lines per page, but if we did that, then the paper would not be ejected correctly at the end of the page, so we shall leave the bottom margin setting at its default value of 3, which is ideal for our printer. Protext will automatically allow for these three lines when calculating how many lines of text to print and then eject the paper at the end correctly.

The final thing that we must do is set printing to single sheet stationery and the methods used will vary with different printers. Ours is fully 'Epson' compatible, so we can make use of the PS (paper sensing) command to control printing and don't need to use the CP OFF command, which we would otherwise have used to set it for single sheet printing.

That now completes all the necessary settings, but as we are going to use plain paper we can save ourselves some repetitive typing by including our address in our template file.

Our template file will therefore look something like:-

```
>CO A5 letter template.
>---!----------------------------------------R
>PL 42       ; page length 42
>TM 0        ; top margin zero
>SM 4        ; side margin 4
>PS ON       ; paper sensing on for single sheet
                                    Lowtech Ltd.
                                    Texport House,
                                    Northern Road,
                                    Glatton,
                                    Cambs.
```

We should now save this template onto our text file disc. We shall save it with the name 'A5LETTER' and in future, every time we want to type an A5 letter, all we have to do is type 'M A5LETTER' before commencing the document. You will notice that we have used the 'M' command to merge the template file in. We could have used the LOAD command instead, but the advantage of using 'M' is that it does not name the document, so there is no chance of us accidentally saving it with the name 'A5LETTER', which could occur if we loaded it in and forgot to change the name of the document.

We can create as many template files as we like, including stored commands for double line spacing, proportional printing, continuous stationery, or whatever we want.

Writing a letter

We shall assume you have loaded Protext with the start of day disc in the normal fashion and Protext is in command mode, waiting for a command. As we are going to write a short letter on single sheet A5 stationery, we will load the appropriate template file, using 'M A5LETTER'. After a second our template will appear in the upper part of the screen and we can press the ESC key to go into edit mode. Once in edit mode, the cursor will be positioned at the start of the document, so pressing CTRL-] will take us to the end of the template and we can now start typing.

Once we have created and corrected the text, it is a good idea to have a thorough read through before printing, to make sure that we have made no mistakes, so we shall use CTRL-[to go to the start of the document and scan through it using CTRL-↓ to make sure.

If the document is likely to be more than one page long, it is also a good idea to make sure that the page breaks fall in convenient places in the text and this can be done very easily by using the CTRL-(and CTRL-) commands, so we shall once again go to the start of the document with CTRL-[and then using CTRL-) move through the document a page at a time. If we find that a page breaks in an unsuitable place, we have only to move back in the text to a suitable place and insert a '>PA' stored command, to force the page to break where we want.

Once we have checked the page breaks, we shall return to command mode with the ESC key and save the document. We type 'S' and press RETURN. We can then enter the name we want to give the letter and press RETURN again. We should try to give it a meaningful name, so that it will be obvious what the contents of each file are, even at a later date.

We shall now carry out a spelling check of the complete document, by typing 'SPELL' from command mode. We insert the dictionary disc (unless using a hard disc) as and when told to and make whatever corrections are necessary as the checking proceeds. When completed, we are back in command mode and perhaps we were warned that the document may need re-formatting because we altered the length of a word whilst correcting them, so we type FORMAT and then resave the document by typing 'S' and pressing RETURN twice.

Alternatively, we could have checked the spelling in the document as we created it, either using CTRL-Q to check a single word, or CTRL-S to check the remainder of the document from the cursor position.

We shall print two copies of the letter, a draft one for our records and a final one in Near Letter Quality. Having made sure that we have put a piece of paper into the printer and that it is positioned correctly, all we need to do is type 'P' (the abbreviation for the PRINT command) and press RETURN and our draft copy will be printed. If the letter is more than one page long, the printer will stop at the end of the first page and allow us to insert a new sheet before pressing the 'on line' button on the printer to print the next page. Once that is done, we can insert another new piece of paper and this time we shall use the 'PQ' command (PRINTQ) to print the second copy in NLQ mode.

That is all there is to it. It has probably taken as long to read this chapter as it would have done to write the letter! If you now want to write another letter, you can type CLEAR from command mode to remove the old letter from memory. There are, of course, many other printing commands available, but for most purposes, these will be the ones that you will use most.

A6. Using a RAM disc

(a) PC

*This appendix describes a method for improving the performance of Protext on a floppy disc machine by using a RAM disc (or 'virtual disc'). First, the procedure for setting up a RAM drive is described. Second, it is explained how to put COMMAND.COM onto the RAM drive. This will enable DOS commands and batch files to be executed if the STARTUP disc has been removed from drive A. The commands affected include COPY, TIME, DATE, and *DIR. Third, the procedure for using the spelling checker with the dictionary on the RAM drive will be explained. Note that at least 640K must be present to take advantage of this.*

A startup disc should have been created previously (see 'Getting Started').

Note: Using a RAM disc on a hard disc system

The only recommended use of this is on AT systems with 2M of memory or more, when a large RAM disc may be set up and used for the temporary text path. CONFIG should be used to change the temporary text path to D:\ (replace D with the RAM disc letter, if different). Allow more than twice the size of the largest file to be edited.

This will make editing very large files noticeably quicker.

1. To set up a RAM drive

The file CONFIG.SYS on the Protext Startup disc will need to be altered. Run Protext and load the file CONFIG.SYS. A line will need to be added at the end of the file. The exact form varies between different computers.

For the IBM PC the line is: DEVICE = \VDISK.SYS 30

For MS-DOS the line is: DEVICE = \RAMDRIVE.SYS 30

Users of DOS v4.0 or above should replace the figure 30 by 40.

If the dictionary is to be copied onto the RAM drive either expanded or extended memory will be needed and a RAM drive of at least 375K should be set up (using the /A or /E option). Consult the MS-DOS manual for details.

For other computers the line required will probably be one of the above. To discover which, put the DOS master disc in drive A and type:

```
DIR A:*.SYS
```

If one of RAMDRIVE.SYS or VDISK.SYS is listed then use the corresponding form. If neither is listed then refer to the operating instructions supplied with the computer to ascertain how to define a RAM drive. The file RAMDRIVE.SYS or VDISK.SYS, as appropriate, should be copied onto the STARTUP disc, from the DOS master disc supplied with your computer. To do this, put the DOS master disc in drive A. If you have 2 drives, put the STARTUP disc in drive B. Type:

```
COPY A:RAMDRIVE.SYS B:                (or COPY A:VDISK.SYS B:)
```

The RAM drive will not become effective until you restart the computer using the STARTUP disc.

2. Copying COMMAND.COM to the RAM drive

The file AUTOEXEC.BAT on the STARTUP disc needs to be amended so that COMMAND.COM is copied to the RAM drive. Load the file AUTOEXEC.BAT and add the following two lines at the start of the file. These lines must appear before the line that loads Protext.

```
COPY COMMAND.COM C:
SET COMSPEC=C:\COMMAND.COM
```

3. Copying the dictionary to the RAM drive.

Note: 1M memory is needed. Remember to set up a 375K RAM drive.

(a) 1 and 2 drive systems

Protext must now be configured so that it looks at the dictionary on drive C. Type 'CONFIG' to run the configuration program. Press 'M' and 'S' to change the spelling checker options. Edit the first two dictionary names by adding 'C:\' to the start. Edit the quick dictionary name by adding 'C:\' to the start. Do not change the 'USER.LEX' entries. The first dictionary name should then be 'C:\ARNORENG.LEX', the second 'C:\ARNORE2.LEX' and the quick dictionary should be 'C:\ARNORENG.QIC'.

(b) 2 drive system only

Add the following lines to the file AUTOEXEC.BAT on the STARTUP disc, before the line that loads Protext:

```
COPY B:\ARNORENG.* C:
COPY B:\ARNORE2.LEX C:
```

When starting up, place the DICTIONARY disc in drive B.

(c) 1 drive system only

After Protext has been loaded from the STARTUP disc, put the DICTIONARY disc in the drive and type the following commands, which will copy the dictionary to the RAM drive:

```
COPY \ARNORENG.* C:
COPY \ARNORE2.LEX C:
```

Note: Adding words to the dictionary

The set up described above will only copy the main dictionary to the RAM drive. Words that are added are stored in the user dictionary on the DICTIONARY disc. Thus it is not necessary to copy the dictionary back after using Protext.

(b) Atari ST

On machines with at least 1M of memory a RAM drive may be used with Protext to speed up spell checking enormously. The most convenient way to use Protext with a RAM drive is to copy your RAM drive software onto the Protext Startup disc. It may be possible to store this in the \AUTO folder so that the RAM disc is automatically installed on startup.

Warning: *Some ST RAM disc software does not correctly emulate all of the disc filing system commands and may cause Protext to crash with a 'fatal system error' while spell checking. If this occurs then the RAM disc software being used is not suitable for use with Protext.*

This version of Protext is supplied with a RAM disc program. See the 'README' file for details of installation and use.

The following assumes that the RAM disc is drive C. If not replace 'C:' with 'D:' or the appropriate letter throughout.

Protext must be configured so that it looks at the dictionary on drive C. Type 'CONFIG' to run the configuration program. Press 'M' and 'S' to change the spelling checker options. Edit the first two dictionary names by adding 'C:\' to the start. Edit the quick dictionary name by adding 'C:\' to the start. Do not change the 'USER.LEX' entries. The first dictionary name should then be 'C:\ARNORE1.LEX', the second 'C:\ARNORE2.LEX' and the quick dictionary should be 'C:\ARNORENG.QIC'.

How to copy the dictionaries to the RAM disc

The following lines should be added to the file 'EXFILE' on the Protext Program disc. If there is no file called 'EXFILE' then create one with the lines:

```
UPDATE ARNORE1.LEX C:
UPDATE ARNORENG.QIC C:
UPDATE ARNORE2.LEX C:
```

Note: Smaller dictionary

The dictionary 'ARNORE1.LEX' used here is a smaller version of the main dictionary. It should be used on a 1M machine as it occupies less space. This is an 80,000 word dictionary without the legal and medical supplements. To use the full dictionary replace 'ARNORE1.LEX' by 'ARNORENG.LEX' in the above.

Note: Adding words to the dictionary

The set up described above will only copy the main dictionary to the RAM drive. Words that are added are stored in the user dictionary on the DICTIONARY disc. Thus it is not necessary to copy the dictionary back after using Protext.

(c) Amiga

On machines with at least 1M of memory (i.e. not A500, unless extra memory has been added), a RAM drive may be used with Protext. This increases the speed of spell checking from edit mode (that is the CTRL-S and CTRL-Q commands and the spell check whilst typing facility). Using a RAM disc is recommended only if these commands are being used - otherwise a better use of memory is to increase the space allocated to the text.

The examples shown use the device name 'RAM:' to represent the RAM disc. Depending on the particular RAM disc system being used this may need to be changed to 'RAD:' or 'VD0:', for example.

Protext must be configured so that it looks at the dictionary on RAM:. Type 'CONFIG' to run the configuration program. Press 'M' and 'S' to change the spelling checker options. Edit the first two dictionary names by adding 'RAM:' to the start. Edit the quick dictionary name by adding 'RAM:' to the start. Do not change the 'USER.LEX' entries. The first dictionary name should then be 'RAM:ARNORE1.LEX', the second 'RAM:ARNORE2.LEX' and the quick dictionary should be 'RAM:ARNORENG.QIC'.

How to copy the dictionaries to the RAM disc

The following lines must be present in the 'S/STARTUP-SEQUENCE' file on the Protext Program disc:

```
COPY PROTEXT-DICTIONARY-DISC:ARNORE1.LEX RAM:
COPY PROTEXT-DICTIONARY-DISC:ARNORENG.QIC RAM:
COPY PROTEXT-DICTIONARY-DISC:ARNORE2.LEX RAM:
```

To add these lines, put the Protext Program Disc in df0: and with Protext in command mode type:

```
LOAD S/STARTUP-SEQUENCE
```

The 'COPY' lines will be seen near the start of the file with semicolons at the beginning. Remove the semicolons and save the file. The dictionaries will now be copied to the RAM disc whenever the Protext Program Disc is used to boot up.

Note: Smaller dictionary

The dictionary 'ARNORE1.LEX' used here is a smaller version of the main dictionary. It should be used on a 1M machine as it occupies less space. This is an 80,000 word dictionary without the legal and medical supplements. To use the full dictionary replace 'ARNORE1.LEX' by 'ARNORENG.LEX' in the above.

Note: Adding words to the dictionary

The set up described above will only copy the main dictionary to the RAM drive. Words that are added are stored in the user dictionary on the DICTIONARY disc. Thus it is not necessary to copy the dictionary back after using Protext.

A7. Protext key tokens

These are the token values used by Protext. The list shows the key that returns the token (unless it has been redefined with CONFIG) and a description of the function. These are the numbers that are found in the macro file 'PROTEXT.KEY'.

Key:

^	Control
s	Shift
a	Alt
(..)	token assigned to an expansion token

token	**key**	**description**
0-255		extended ASCII character set (particular to environment)

But with these exceptions:

token	key	description
0		cannot be used
5		cannot be used
9	TAB	insert tab
13	RET	carriage return

On the Amiga (and PC, as these are the normal character codes):

token	key	description
24	a↑	↑ symbol
25	a↓	↓ symbol
26	a→	→ symbol
27	a←	← symbol
256		unallocated expansion token
257-266	f1-f10	expansion tokens
267-276	sf1-sf10	expansion tokens
277-286	^f1-^f10	expansion tokens
287-296	af1-af10	expansion tokens
297-322	aA-aZ	expansion tokens
323-328	saA-saZ	expansion tokens

token	key	description
349	f11	expansion token
350	sf11	expansion token
351	^f11	expansion token
352	af11	expansion token
353	f12	expansion token
354	sf12	expansion token
355	^f12	expansion token
356	af12	expansion token
357-382		unallocated expansion tokens
383	HELP	expansion token (ST/Amiga, unallocated on PC)
512	^@	set or goto marker
513	^A	swap adjacent characters
514	^B	box mode switch
515	^C	centre line
516	^D	default ruler
517	^E	delete to end of line
518	^F	format to end of paragraph
519	^G	goto line, page or column
520	^H	
521	^I	insert line
522	^J	justify on/off
523	^K	clear block markets
524	^L	goto last position
525	^M	move block
526	^N	non break character
527	^O	copy block from other file
528	^P	page mode on/off
529	^Q	spell check single word
530	^R	ruler line
531	^S	spell check from cursor
532	^T	reserved
533	^U	undelete block
534	^V	view (various options)
535	^W	word wrap on/off
536	^X	enter printer control code
537	^Y	swap to other file
538	^Z	set block marker

token	key	description
539	^[	move to start of memory
540	^\	lower case
541	^]	move to end of memory
545	Del	delete forwards
601	(af1)	swap to document 1
602	(af2)	swap to document 2
603	(af3)	swap to document 3
604	(af4)	swap to document 4
605	(af5)	swap to document 5
606	(af6)	swap to document 6
607	(af7)	swap to document 7
608	(af8)	swap to document 8
609	(af9)	swap to document 9
610	(af10)	swap to document 10
611	(af11)	swap to document 11
612	(af12)	swap to document 12
639	←DEL	delete backwards
671		ignored
689	s^(	move to start of column
690	s^)	move to next column
691		start addition
692		end addition
693		insert result of addition
694	aTAB	indent tab
695	s^Y	physically swap top and bottom files
697	s^X	split/unsplit screen
699	aG	graphics/text mode switch (PC only)
700	s^/	upper case word
701	s^\	lower case word
702		top of screen
703		bottom of screen
705	a\	grave accent
706	a~	tilde accent
707	a"	diaeresis/umlaut accent

token	key	description
708	sa^	caron accent
709	a%	ring accent
710	a^	circumflex accent
711	a'	acute accent
712	sa'	double acute accent
713	sa\	double grave accent
714	a_	macron accent
715	sa~	breve accent
716	sa.	overdot accent
717	sa,	cedilla accent
718	a;	ogonek accent
720	s^F	format whole paragraph
722		clear text
723	s←DEL	delete word backwards
724	^←DEL	delete to start of line
725	^Home	start of text
726	^End	end of text
727	^kpd+	next marker
728	^kpd-	previous marker
729	(f5)	previous find
730	^<	back paragraph
731	^>	forward paragraph
732	^(	back page
733	^)	forward page
734	^8	split line
735	^=	join lines
736	(f6)	next find
737	^TAB	insert mode on/off
738		
739	^-	soft hyphen
740	sTAB	move to next tab
741	sDEL	delete word right
742	(^f3)	delete line
743	^/	upper case
744	^DEL	delete block
746	(f10)	copy block
747	^space	insert space
748	sRET	move to left margin on next line

token	key	description
752	↑	cursor up
753	↓	cursor down
754	←	cursor left
755	→	cursor right
756	s↑	scroll up one line
757	s↓	scroll down one line
758	s←	move word left
759	s→	move word right
760	^↑	scroll up one screenful
761	^↓	scroll down one screenful
762	^←	move to start of line
763	^→	move to end of line
764	Esc	escape
765		goto edit mode
766		goto command mode
768		memory editor
769		keyboard driver switch (PC/ST)
772		doc/prog mode switch
773		Gem desktop (ST)
774	(^f1)	record macro
775	s^S	spell check whilst typing on/off
776	(^f7)	line drawing mode
777	(^f8)	line drawing with character
778	s^I	auto indent mode switch
779	s^R	auto reformat switch
780		click left mouse button
781	(aF)	43/50 line display switch
785	s^E	delete to end of sentence
786	s^f6	find word at cursor forwards
787	s^f5	find word at cursor backwards
788	(aD)	current date
789	(aT)	current time
790	^HELP	scroll lock (ST/Amiga)
791		num lock (ST/Amiga)
792	s^←	move to start of sentence
793	s^→	move to end of sentence
794		double click

token	key	description
801	(saf1)	select keyboard language 1
802	(saf2)	select keyboard language 2
803	(saf3)	select keyboard language 3
804	(saf4)	select keyboard language 4
805	(saf5)	select keyboard language 5
806	(saf6)	select keyboard language 6
807	(saf7)	select keyboard language 7
808	(saf8)	select keyboard language 8
809	(saf9)	select keyboard language 9
810	(saf10)	select keyboard language 10
826		dead acute accent
827		dead grave accent
828		dead diaeresis/umlaut accent
829		dead circumflex accent
830		dead ring accent
902	(f3)	menu key
903	s^←DEL	delete to start of sentence
904		sideways scroll left
905		sideways scroll right
906	a=	spell ignore marker
907		link files
908		menu mouse button

A8. IBM PC Character Set

	DEC	0	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240
DEC	HEX	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
0	0		►	SP	0	@	P	`	p	Ç	É	á	░	└	╨	α	≡
1	1	☺	◄	!	1	A	Q	a	q	ü	æ	í	Đ	┴	╤	ß	±
2	2	☻	↕	"	2	B	R	b	r	é	Æ	ó	ð	┬	╥	Γ	≥
3	3	♥	‼	#	3	C	S	c	s	â	ô	ú	│	├	╙	π	≤
4	4	♦	¶	$	4	D	T	d	t	ä	ö	ñ	┤	─	╘	Σ	⌠
5	5	♣	§	%	5	E	U	e	u	à	ò	Ñ	╡	┼	╒	σ	⌡
6	6	♠	▬	&	6	F	V	f	v	å	û	ª	╢	╞	╓	µ	÷
7	7		↨	'	7	G	W	g	w	ç	ù	º	╖	╟	╫	τ	≈
8	8		↑	(	8	H	X	h	x	ê	ÿ	¿	╕	╚	╪	Φ	°
9	9	TAB	↓	)	9	I	Y	i	y	ë	Ö	⌐	╣	╔	┘	θ	•
10	A	LF	→	*	:	J	Z	j	z	è	Ü	¬	║	╩	┌	Ω	•
11	B	♂	←	+	;	K	[	k	{	ï	ø	½	╗	╦	Ł	δ	√
12	C		∟	,	<	L	\	l	¦	î	£	¼	╝	╠	ł	∞	ⁿ
13	D	CR	↔	-	=	M	]	m	}	ì	Ø	¡	╜	═	Ŀ	φ	Þ
14	E	♫	▲	.	>	N	^	n	~	Ä	₧	«	╛	╬	ŀ	∈	þ
15	F	☼	▼	/	?	O	_	o		Å	ƒ	»	┐	╧	ı	∩	

Note: ø, Ø, ł, Ł, ŀ, Ŀ, ı, ð, Đ, Þ and þ are not part of the normal character set, but are defined by Protext. They are only displayed in graphics modes.

A9. ST Character Set

	DEC	0	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240
DEC	HEX	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
0	0			SP	0	@	P	`	p	Ç	É	á		└	╨	α	≡
1	1			!	1	A	Q	a	q	ü	æ	í	Đ	┴	╤	ß	±
2	2			"	2	B	R	b	r	é	Æ	ó	ð	┬	╥	Γ	≥
3	3			#	3	C	S	c	s	â	ô	ú	│	├	╙	π	≤
4	4			$	4	D	T	d	t	ä	ö	ñ	┤	─	╘	Σ	⌠
5	5			%	5	E	U	e	u	à	ò	Ñ	╡	┼	╒	σ	⌡
6	6			&	6	F	V	f	v	å	û	ª	╢	╞	╓	µ	÷
7	7			’	7	G	W	g	w	ç	ù	º	╖	╟	╫	τ	≈
8	8		↑	(	8	H	X	h	x	ê	ÿ	¿	╕	╚	╪	Φ	°
9	9	TAB	↓	)	9	I	Y	i	y	ë	Ö	⌐	╣	╔	┘	θ	•
10	A	LF	→	*	:	J	Z	j	z	è	Ü	¬	║	╩	┌	Ω	•
11	B		←	+	;	K	[	k	{	ï	ø	½	╗	╦	Ł	δ	√
12	C			,	<	L	\	l	¦	î	£	¼	╝	╠	ł		ⁿ
13	D	CR		-	=	M	]	m	}	ì	Ø	¡	╜	═	Ŀ	φ	Þ
14	E			.	>	N	^	n	~	Ä		«	╛	╬	ŀ	∈	þ
15	F			/	?	O	_	o		Å	ƒ	»	┐	╧	ı	∩	

Note: The line draw characters (hex B3-DA), ø, Ø, ł, Ł, ŀ, Ŀ, ı, ð, Đ, Þ and þ are not part of the normal ST character set, but are defined by Protext.

A10. Amiga and Archimedes Character Set

	DEC	0	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240
DEC	HEX	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
0	0			SP	0	@	P	`	p		ı		°	À	Đ	à	đ
1	1			!	1	A	Q	a	q		Ŀ	¡	±	Á	Ñ	á	ñ
2	2			"	2	B	R	b	r		ŀ	¢	²	Â	Ò	â	ò
3	3			#	3	C	S	c	s		Ł	£	³	Ã	Ó	ã	ó
4	4			$	4	D	T	d	t		ł	¤	´	Ä	Ô	ä	ô
5	5			%	5	E	U	e	u		│	¥	µ	Å	Õ	å	õ
6	6			&	6	F	V	f	v		┤	¦	¶	Æ	Ö	æ	ö
7	7			'	7	G	W	g	w		┐	§	•	Ç	×	ç	÷
8	8		↑	(	8	H	X	h	x		└	¨	¸	È	Ø	è	ø
9	9	TAB	↓	)	9	I	Y	i	y		┴	©	¹	É	Ù	é	ù
10	A	LF	→	*	:	J	Z	j	z		┬	ª	º	Ê	Ú	ê	ú
11	B		←	+	;	K	[	k	{		├	«	»	Ë	Û	ë	û
12	C			,	<	L	\	l	|		─	¬	¼	Ì	Ü	ì	ü
13	D	CR		-	=	M	]	m	}		┼	-	½	Í	Ý	í	ý
14	E			.	>	N	^	n	~		┘	®	¾	Î	Þ	î	þ
15	F			/	?	O	_	o			┌	¯	¿	Ï	ß	ï	ÿ

Note: The arrows (hex 18-1B), line draw characters (hex 95-9F), ł, Ł, ŀ, Ŀ and ı are not part of the normal character set, but are defined in the Protext program.

A11. Glossary

Alert box	A box that pops up on the screen to give information, often an error message.
ASCII	A code used to represent text within a computer.
Backup file	The copy of the previous version of a file that is retained whenever a file is saved.
Block	A contiguous portion of text defined by two markers at the first and last character included.
Byte	The unit of storage of a computer capable of holding a single character.
Caps Lock	The state of the computer in which all letters typed are entered as upper case.
Catalogue	A list of files stored on a disc.
Character	Any letter, digit or symbol.
Clicking	Pressing a mouse button, usually the left button.
Command	An instruction to the computer to do something.
Command button	A box which, when clicked on, causes an action to occur or be cancelled (e.g. the 'GO' and 'Cancel' buttons in the print dialogue box).
Command Mode	The mode of operation of a computer program (e.g. PROTEXT, BASIC) that allows commands to be entered.
Configuration	The operation which customises the word processor for a particular printer or an individual user.
Control code	A number that represents an instruction and not a character. In particular an instruction to a printer (e.g. to underline or print italics).

CP/M	An old operating system. The forerunner to MS-DOS.
Cursor	The blob on the screen marking the place at which text will be entered.
Database	A program to store and display information. An electronic card index.
Data files	Files created from the information in a database or created with a word processor, frequently for use in mail merging.
Default	The initial setting of some feature of the word processor, such as insert mode on, page length 66.
Desktop	The initial screen on the Atari ST, from which programs are run by double-clicking with the mouse.
Dialogue box	A box which appears on the screen to solicit information from the user. It may contain one or more text boxes, command buttons and option buttons.
Directory	A collection of files on a disc. Particularly on large capacity discs such as hard discs it is convenient to separate the files into different directories. The catalogue of files only shows one directory at a time.
Disc	A means of magnetic storage of data allowing fast access.
Document mode	The usual mode of operation of PROTEXT, in which all the formatting features may be used.
DOS	Another name for MS-DOS or PC-DOS.
Double-click	To press a mouse button (usually the left button) twice in quick succession.
Drawer	The same as directory (Amiga terminology).
Drive	Disc drive, the part of the computer into which discs are inserted.

Drop down menu	The type of menu used in Protext, that drops down from a menu bar at the top of the screen.
Edit mode	The mode of operation of a word processor that allows text to be entered and changed.
Expansion token	A special character which represents a string of characters. The means by which macros work.
File	A group of data collected together and stored on disc.
File requester	The same as file selector (Amiga terminology).
File selector	A special dialogue box that allows a file to be chosen for some operation.
Floppy disc	A type of disc commonly used with microcomputers.
Folder	The same as directory (Gem terminology).
Font	One of the character sets available on a printer.
Footer	One or more lines printed at the bottom of each page, in which titles and page numbers can be printed.
Form feed	A control code that causes a printer to feed the paper through to the start of the next page.
Format	(i) An operation on a portion of text which causes the text to be arranged in some regular way. (ii) An operation that must be performed on a disc to make it ready for use.
Function keys	Special keys on the keyboard, usually configurable to carry out special tasks.
Gadget	A collective name for any of the boxes or buttons that can appear in a dialogue box (Amiga terminology).
GEM	The operating system used on the Atari ST.
Global	Everywhere, i.e. throughout the entire text.

Hard disc	A fast, high capacity disc drive.
Header	One or more lines printed at the top of each page, in which titles and page numbers can be printed.
K	A unit of storage equal to 1024 bytes.
Line feed	A control code that causes a printer to feed the paper through by a single line.
List box	A box used to select from a list of items which may be scrolled up and down to reveal further items.
Load	Copy data from disc to the memory so that the data may be used.
Lower case	Small letters.
Macro	A sequence of key presses reproduced by an abbreviated command such as pressing a function key.
Mail merging	An operation whereby a standard letter or document is printed many times using names and addresses taken from a file.
Margin	(i) The normal limit of text to the left and right. (ii) The blank space on the paper surrounding the text.
Marker	A pointer to a particular location in the text.
Memory	The part of a computer where data (programs, text, etc.) is stored, and may be accessed quickly.
Merge	Load a file into memory without clearing memory first.
Mode	One of a number of states of operation of a program, e.g. edit or command mode, insert or overwrite mode.
Mouse	A means of giving commands to a computer as an alternative to the keyboard.
MS-DOS	The operating system used on most PC computers.

Operating system	A special program supplied with a computer which controls use of the disc drives, screen and keyboard.
Option button	A box which, when clicked on, is highlighted to indicate the selection of an option.
Page throw	An instruction to start a new page when printing.
Path	A directory name or a sequence of directory names which identify where a file is stored.
PC-DOS	The operating system used on IBM PC computers. It is virtually the same as MS-DOS.
Printer driver	A file that configures a word processor to print on a particular printer.
Program mode	The alternative to document mode, used for editing programs. The formatting features are not available.
Prodata	A database program produced by Arnor and particularly suitable for use with Protext.
Protext	A word processor program produced by Arnor.
Prowort	The German version of Protext.
Pull down menu	The same as drop down menu.
Requester	The same as dialogue box (Amiga terminology).
Return	(i) The code used in the computer a line end. (ii) A key on the keyboard used to terminate a line.
Right justification	An operation on a line of text which causes the text to be spread out over the full length of the line.
Ruler	A line at the top of the screen showing left and right margins and tabs.
Save	Copy data from the memory of the computer to a disc so that the data may be preserved.

Shift Lock	The state of the computer in which all keys pressed act as if the key is pressed together with SHIFT.
Soft character	A character in the text used for formatting.
Status line	The line at the top of the screen when in edit mode which shows various useful information.
Stored command	A command concerned with printing, that is stored in the text but is not printed as part of the text.
String	A sequence of characters which may be letters, numbers, spaces, punctuation marks, etc.
Tab	A position in a line to which the cursor moves when the TAB key is pressed.
Text box	A box in which text is entered and edited.
Toggle	A command which switches some function on and off.
Token	A number that can be assigned to a key to produce a certain effect when the key is pressed.
Upper case	Capital letters.
Utility	A program or command, distinct from the main program which performs some useful function.
Wildcard	A special character which will match any character when searching for a string.
Word processor	A computer program that handles the entering, storing, formatting and printing of text.
Word wrap	The operation of moving a word onto the next line when there is not enough room on the current line.
Workbench	The Amiga operating system program which displays icons and allows more than one program to be run simultaneously by double-clicking with the mouse.

Index

A 154
A4 200
AB 154, 301
ABANDON 154
Abandon code sequence 154
Abbreviations 121, 145, 154
Abs 231
ACC 155
Accent 306
Accents 92, 282
ACCESS 155
Active ruler line 76
Acute 92
ACUTE 306
Address labels 250
Addresses 235, 243
Alert box 364
Alpha 94
ALT 48
Ambiguous filenames 150-151
Amiga 14, 296
Amiga and Archimedes Character Set 363
AmigaDOS 151-153
Ampersand 241
AN 155
ANAGRAM 155
And 232
Apostrophes 121
Appending to file 260
Arithmetic 271
ARP file requester 35
Arrow 93
ASCEND 307
ASCII 235, 296, 364
ASCII file 186
Ask for variable 247
Assemblers 183
AT 14, 169
Atari 14
Attributes 64
Audible warning 282
Auto indent 99, 282
Auto re-formatting 279
Autoexec file 280
AUTOEXEC.BAT 276
Automatic sace 96
Automatic save maximum size 280
Automatic save time period 280
AV 204, 247
Average proportional width 119

B 156
BACK 156
Back gadget 152-153
Background colour 281
Background printing 110, 156, 159, 282, 318
BACKSPACE 48
Backspacing 104
Backup files 148, 364
Backups when saving 282
Backwards search 87
Banner line 69
BASIC programs 176
Beta 94
Binary file 214
Blank lines 239, 316
Block 73, 108, 160, 166, 176, 178, 186, 364
  copying 73
  counting 160
  defining 72
  deleting 73
  fixing 166
  formatting 166
  moving 73
  numbering 176
  printing 108, 178
  saving 186
  undeleting 73
Block marker 72, 74
Block markers 97
BM 204
Bold 98, 104-105
Bottom margin 204, 279
Box 74-75
  copying 74
  defining 74
  deleting 75
  moving 74
Box mode 74
Breve 92
BREVE 306
BU 156
BUILD 156
Building dictionary 156
Byte 364

C 99, 157, 191
CA 157
CALC 157
Calculations 271
Calculator 157
Caps Lock 55, 364
Caron 92
CARON 306
Case 89
Case specific 87

CAT 70, 149, 157
Catalogue 364
Catalogue files 157
CC 205, 301
CD 146, 149, 152, 158
CE 205
Cedilla 92-93
CEDILLA 306
Centre Tabs 79
Centring text 85, 205
Centronics 176
CF 205, 242, 256
CFG 159
CGA 98
Change spelling 126
Changing discs 149
Changing drives 65
Character 364
  defining 190
Character pitch 106
Character widths 206, 223, 315
CHDIR 158
Chevron 76
Choose character window 93
Circumflex 92
CIRCUMFLEX 306
CLEAR 70, 158
Clear block markers 74
Clear screen 206
Clearing the text 70
CLEARKEYS 158
CLI 152-153
Clicking 59, 364
Clock 282
Closing data files 256
Closing write file 261
CLRK 158
CLS 159
CO 159, 205
COD 160
Code 141
CODE 159
Coercion of types 233
Colours 168, 281
Column 60, 167
Columns 220
Comma 79
Comma delimited 236, 238-239
Command 364
Command button 364
Command line options 287
Command mode 47, 50, 69, 144, 282, 364
Command prompt 69, 149
Command recall 145
Commands 154
Comments 82, 204-205
Compilers 183
Condensed 105-106, 111, 322
Condensed elite 106
Condensed pica 106
Conditional printing 251
Conditions 251, 270
CONFIG 73, 79, 130, 143, 146, 159, 195, 276
Configuration 172, 190, 276, 278, 364
Configuration files 276
Configure 177, 187
CONFIG.SYS 276
CONT 159
Contents 3, 136
  table of 132
Continuous printing 206, 279
Control code 364
Control codes 109, 279, 364
Control panel 318
Conventions 49
Conversion of files 295
CONVERT 159, 295
Copies
COPIES 301
  number of 218
COPY 152, 159
COPYCC 308
Copying a block 73
Copying a box 74
Copying files 66
COPYRC 308
Correcting text 54
COUNT 70, 160
COUNTB 160
COUNTD 160
Counting block 160
Counting words 70, 160
CP 206, 301, 365
CPC 296
CPD files 311
CPM 296
Creating a new directory 65
Crossdos 296
Crosswords 155, 165
CS 206
CT 160
CTB 160
CTRL 48
CTRL/ALT/SHIFT lock 281
Current directory 146
Current disc 149
Current filename 146, 176, 185
Cursor 55, 167, 365
  moving 167
Cursor flash 281
Cursor keys 55
Cursor movement 58

Cursor type 281
Cut and paste 72, 74
CV 159
CW 206, 308

D 161
DACUTE 306
Daisywheel printer 104, 222
Danish 94
Data file 242-243
Data files 207, 235, 237, 240-242, 256, 260, 365
Database 245, 365
Databases 235, 238, 240
Date 96, 273
DATE 161
Date format 282
Datenum 231
Day 273
Dayofweek 231
DC 206, 271
DE 206
Debugging 207
Decimal character 206, 271, 282
Decimal places 207, 272
Decimal tab 78-79, 241
Default 365
Default font 116, 119, 210
Default printer driver 104
Default ruler 77
Defining a block 72
Defining a block with the mouse 73
Defining a box 74
Defining characters 190
DEL 48, 56, 152, 163
Delete line 56
Delete to end of line 56
Delete to end of sentence 56
Delete to start of line 56
Delete word 56
Deleting 56
Deleting a block 73
Deleting a box 75
Deleting a database 66
Deleting files 163
Delta 94
DELWORDS 161
Depth gadget 153
Desktop 365
Desktop accessories 28
DF 207, 242, 256
DGRAVE 306
Diaeresis 92
DIAERESIS 306
Dialogue box 365
Dialogue boxes 44
Dictionaries 147, 173
Dictionary 156, 160-161, 165, 169-170, 175, 280
  building 156
  counting 160
  creating 175
  listing 170
  quick 124, 147, 280
Dictionary disc 124
Dictionary files 123
Dictionary prompt 280
Diphthong 93
DIR 162
Directory 146, 158, 185, 365
  creating 175
  current 146
  parent 158
  removing 185
  root 158
Directory listing 63
Directory Tree 67
Directory tree 146
Disc 108, 365
  current 149
  dictionary 124
Disc free space 67
Disc Utilities 63
Discs
  changing 149
DISKCOPY 152
Display options 281
Displaying messages 263
Div 231
DM 207
DOC 162
Document
  loading 69
  printing 70
  saving 69
  spell checking 124
  swapping 190
Document mode 162, 280, 282, 365
Documents 190
Dollar symbol 236
DOS 365
Dot matrix printer 104
DOUBLE 302
Double acute 92
Double-click 365
Downloading fonts 214
DP 207, 272
DR 162
Draft 104
Dragging 100
Drawer 146, 365
DRIVE 162
Drives 146, 149, 365
Drop down menu 366

DU 207
DW 161

E 163
EA 207
EC 112, 208
ECHO 163
Edit mode 46, 49, 54, 366
Editing 144
Editing options 279
EF 208
EGA 98
EH 208
EI 208, 252
EL 208, 253
Electronic Mail 186
Elite 105, 322
Else 253
ELSE 312
EM 209
Emphasised 105
Empty fields 236, 264
EN 209
END 302
Endif 252
ENDIF 312
Endnotes 202, 209
English 94
Enlarged 105-106
Enlarged titles 85
ENTER 308
Entering text 54-55
Envelopes 193
Environment variable 276
EP 209
Epsilon 94
ERASE 163
Erasing files 66
ESC 69
Escape character 88, 142
Evaluation operators 230
Even margin 200
Even pages 221
EX 209
Exclamation mark 88
EXEC 143, 163
Exec files 138
Exec Files 141
EXECS 164
EXFILE 143, 280
EXIT 308
Expansion tokens 138, 366
Export 238
Expression 157, 229-230, 271
Expression Evaluator 230
EXT 164
Extension 164, 280
External programs 147, 152

F 164
F2 70, 149
F3 44
FB 165
FD 210, 302
FE 210
FF 164, 210, 302
Field 235
File 171, 184, 189, 192, 366
  ASCII 186
  help 147
  inserting 215, 248
  loading 171
  macro 147
  merging 175
  printing 178
  printing to 109, 179
  renaming 184
  splitting 189
  spooling 189
  typing 192
  writing to 227, 260
File and disc options 280
File conversion 295
File Protection 67
File requester 366
File Selector 45, 63
File selector 366
File sizes 157
Filename
  current 146, 176, 185
Filename extension 164, 171
Filenames 147, 150-151
Files 70, 147, 163
  backup 148
  data 235
  deleting 163
  exec 141
FIND 71, 87, 89, 164
Find next 89
Find previous 89
Find word at cursor 90
FINDB 90, 165
Finding text 71, 87
FINDW 165
First page 224
First Word 297
FIX 166
FIXB 166
Fixed pitch 111
Fixing block 166
Fixing text 166, 212
Floppy disc 366

FM 211
FN 211
FO 211-212
Folder 146, 366
FONT 308
Font 366
Footer 211, 366
  even 208
  last page 216
  odd 219
Footer margin 279
Footers 110, 198, 280
  multiple line 199
Footnote 210
Footnote marker 88, 202, 211
Footnotes 202, 211
Form feed 164, 210, 281, 318, 366
FORMAT 152, 166
Format 366
FORMATB 166
Formatting 83-84
Formatting block 166
Formatting text 166
Formatting whilst printing 212
FP 212, 302
Frac 231
French 94
FSORT 167
FT 166
FTB 166
Function keys 366
FW 165
FX 212

G 167
Gadget 152-153, 366
Gamma 94
GC 213
GEM 288, 366
Gem desktop 28
Gem file selector 28
German 94
Getting Started - Amiga 29
Getting Started - Archimedes 36
Getting Started - Atari ST 23
Getting Started - IBM PC and Compatibles 16
GI 213
Global 366
Global search 87
Glossary 364
GOTO 167
Granule 313
Graphics 95, 214
Graphics mode 53, 99
Grave 92
GRAVE 306
Greater than symbol 76
Greek 94
GROUP 302

H 167
Half 93
Half lines 215
Hard disc 367
HE 213
Header 213, 322, 367
  even 208
  odd 219
Header margin 199, 279
Headers 110, 198, 280
  multiple line 199
Help 54, 144
HELP 167
Help file 147
Hercules 98
Hints and tips 316
HL 303
HM 214
Horizontal scrolling 62
Hyphen 88
Hyphens 85-86, 88, 97
  non-break 86
  soft 85

IB 214
IBM 15
IBM PC 14
IBM PC Character Set 361
Icon 153
Icons 35
ID 214, 257
IE 214, 257
IF 214, 252-253, 270, 312
If defined 257
If exhausted 257
If undefined 257
Ignore word 127
Ignored words buffer size 280
IN 215, 249, 251, 265
In 232
Indent 99, 282
Indent Tabs 80
Indentation 81
Index 132
INDEX 168
Index markers 61
Index phrase marker 88, 132
Index word marker 88, 132
Indexing 61
INK 168
Input variable 247
INS 48

Insert file 215, 249
Insert line 57
Insert mode 57, 279
Inserting 57
Installation 278
INSWORDS 169
Int 231
Inter paragraph space 197
Introduction 13
Intuition 35
Inverted exclamation mark 93
Inverted question mark 93
Invoices 261
IP 215
Isdate 231
Isnum 231
Italian 94
Italic 317
Italics 98, 105
IU 215, 257
IW 169

Joining lines 62
Joining variables 266
Justification 83, 110

K 367
KEY 173
Key delay 281
Key repeat period 281
KEYB 169
Keyboard 48, 169-170, 281
  alternative commands 338
Keyboard driver 281
Keyboard language 281
Keyboard layout 94
Keyboard layout configuration 35
Keyboard options 281
Keyboard problems 320
Keyboard type 281
KEYDEF 169
Kyocera 118-119

L 171
Labels 250
LANG 170
Languages 92, 94, 170
Large files 50, 130
Laser printer 104, 164, 210, 222, 317
LaserJet 119
Last page 199, 207, 216
Layout 76
Layout options 279
LCFG 172
LD 170
Ldate 231
LDK 172
Left margin 80-81, 226
Len 231
LF 216
Line 56, 60, 167
  deleting 56
  inserting 57
  moving 57
Line drawing 95, 317
Line editing 144
Line feed 281, 367
Line graphics 95
Line numbering 176
Line spacing 110, 216, 279
Lines 176
LINK 170
LISTD 170
Listing the files 70
LK 173
LM 171
LMACROS 171
LOAD 69, 146, 171
Load 367
Load configuration 278
LOADCFG 172
Loading a document 69
Loading file 171
LOADKEYS 140, 172
Look up correct spelling 127
LOOKUP 173
Looping 258
Lower case 55, 367
LS 216
Lwr 231

M 175
Macro 92, 158, 172, 367
MACRO 140, 173
Macro file 147
Macro record mode 138
Macron 92
MACRON 307
Macros 138, 140, 187
  defining 138, 173
  discarding 174
  listing 171
  loading 172
  saving 140, 187
Mail merging 85, 234, 367
  conditional 251
  sample documents 340
MAKED 175
Map 67
Margin 76, 80-81, 110, 199, 204, 279, 367
  bottom 204
  even 200, 209

  footer 211
  header 214
  left 226
  odd 200, 220
  side 226
  top 227
  zeroing 228
Marginal comments 82
Marker 59, 74, 88, 130, 367
  block 97
  footnote 202, 211
  place 97
Markers 97
Maximum file memory size 280
MC 217, 303
MCGA 98
MD 152, 175
Memory 367
Memory options 280
Memory size 280
Menu 100
MENU 304
  dictionary 131
  print 177
Menu key 44
Menu operation 43
MENUEND 304
MERGE 175
Merge 367
Merging file 175
MF 217, 304
Microspace 313
Microspace code 217
Microspacing 111, 217-218, 313
Microspacing factor 313
MKD 175
MKDIR 175
MM 217, 304
Mod 231
Mode 367
Modula 2 99
Monochrome 98
Month 273
Mouse 59, 367
Moving a block 73
Moving cursor 167
Moving lines 57
MP 217, 304
MS 218, 305
MS-DOS 14, 152, 367
Mu 94
Multiple columns 220
Multiple copies 218
Multiple File Editing 101
Multiple markers 61
Multi-tasking 153

N 176
NAME 146, 176
Names 235, 243
NC 218
Near letter quality 105, 108, 181
Nesting 252-253
New page 220
New page after printing 218, 280
Newspaper column printing 112, 220
NLQ 105, 181
Non-break hyphen 86, 88
Non-break space 86, 88, 97
Normal 105
Norwegian 94
Not 231
Notin 232
NOTIN 251
NP 218
NUM 176
NUMB 176
NUMBER 176
Number of copies 108
NUMBERB 176
Numbering block 176
Numbering lines 176
Numbers 121-122
Numeric 230
Numeric arithmetic 271
Numeric variables 207

OC 219
Odd margin 200
Odd pages 222
OF 219
Ogonek 92
OGONEK 307
OH 219
OM 220
Omega 94
OMIT 305
OP 220
Operating system 368
Operators 157, 230-231
Option button 368
Options 287
Or 232
Overdot 92
OVERDOT 307
Overview of Protext 41, 43
Overwrite mode 58, 279

P 178
PA 220
Page 60, 167
Page break lines 98, 279

Page length 110, 200, 221, 279
Page mode 197, 279
Page number 198, 221
Page numbers 198
Page throw 209, 220, 368
  conditional 220
Pages 108
Pagination 200
Paper sensing 223, 318
PAR 176
Paragraph 84
Paragraph indentation 81
Paragraph numbering 274-275
Paragraphs 215
PARALLEL 176
Parallel printer 154
Parameters 154
Parent directory 64, 158
Pascal 191
Path 147, 280, 368
PATH 177
  search 147
  temporary 147
Pattern 155
PAUSE 143, 177
Pause printing 228
PB 178
PC 14, 16, 112, 220, 296
PC-DOS 14, 368
PE 221
PF 179
PFB 179
Phi 94
Pi 94
Pica 105
Pitch 106
PL 221
Place markers 60, 97
PM 177
PN 221
PO 222
Polymorphic 230
Portuguese 94
Possessives 122
Pound sign 317, 320
PP 180, 222, 305
PPQ 180
PPS 181
PQ 181
PQB 182
PR 178, 222
Preferences 177, 288
Previous ruler 77
PRINT 70, 107, 178
Print dialogue box 107
Print list of pages 108, 180
Print to file 109
Print to screen 109
PRINTB 108, 178
Printer 15, 104
PRINTER 178
Printer buffer 110, 154
Printer control codes 88, 97, 105, 205
Printer driver 104, 109, 176, 178, 222, 281, 301, 306, 313-315, 368
  creating 300
Printer driver options 281
Printer port 281
Printer problems 316
Printer requirements 95
PRINTF 109, 179
PRINTFB 109, 179
Printing 104, 110, 113, 156, 206
  background 110, 159
  conditional 251
  list of files 322
  pausing 228
  proportional 113
Printing block 108, 178
Printing by pages 108
Printing commands 107
Printing file 178
Printing from disc 108
Printing in program mode 111
Printing text 70, 107, 178
Printing to file 109, 179
Printing to screen 109, 182
PRINTOFF 179
PRINTON 180
PRINTP 108, 180
PRINTPQ 108, 180
PRINTPS 181
PRINTQ 108, 181
PRINTQB 108, 182
PRINTS 109, 182, 201, 244
PRINTSB 109, 183
Prodata 368
PROFF 179
PROG 183
Program mode 99, 111, 183, 191, 280, 282, 368
Programs 147, 152, 176
Prompt 69, 149
PRON 180
Proportional 105
Proportional printing 85, 113, 217, 222, 313, 315
Proportional printing and editing 114
PROT 184
PROTECT 184
Protext 13, 43, 368
Protext - Compatibility between versions 337
Prowort 368
PS 14, 150, 182, 223, 244

PSB 183
PSCODE 305
Pull down menu 368
PW 119, 223, 309

Q 184
Question mark 88
Quick dictionary 124, 280
QUIT 71, 184
Quitting Protext 71, 184
Quotes 88, 238

R 184
RAM disc 125
RB 185
RC 223, 305
RD 152, 185
Read variables 242
README 42
Read-only 184
Read-write 155
Record 235
Record separators 239
Recording macros 138
Redefine keyboard layout 281
REGULAR 306
REM 306
REN 152, 184
RENAME 184
Renaming file 184
Renaming files 66
Repeat 258-259
Repetitive looping 258
REPLACE 87, 89, 184
REPLACEB 90, 185
Replacing text 87
Report form 238
Requester 368
Reset code 105, 281
Reset default configuration 278
Resizing window 100
RETURN 55
Return 85, 88, 368
  hard 98
  soft 85
Returns displayed 279
Right 368
Right justification 79, 83, 110, 224, 279
Right margin 79-80
Ring 92
RING 307
RJ 224
RMDIR 185
Roman numerals 121, 275
Root directory 158
Round 232
ROUND 271
RP 224, 258
RU 224, 242, 245
Ruler 76-77, 110, 195, 280, 368
Ruler editing with the mouse 80
RULES 309
RUN 153
RV 224, 242, 245, 264

S 185
SA 224
SAB 186
SAVE 69, 146, 185
Save 368
Save new configuration 278
SAVEA 240
SAVEAB 186
SAVEB 186
SAVEKEYS 140, 187
Saving a document 69
Saving ASCII 186
Saving block 186
Saving macros 140, 187
Saving text 185
SB 186
SC 188, 225
SCB 188
Screen 109, 182
  printing to 109, 182
Screen blink 281
Screen Mode 281
Scripts 82
Scroll lock 150, 281
Scroll lock mode 66
Scrolling 61
Sdate 231
Search 87
Search backwards 87
Search path 147, 280
Searching 87
Selecting a directory 64
Selecting a file 64
Self incrementing variables 274
Sentence 56
Separators 229
SER 187
SERIAL 187
Serial delay 281
Serial interface 187
Serial printer 317
SET 188
Set variable 247
SH 188
Sheet feeder 210, 317
SHELL 152-153, 276
Shift Lock 369

SHOW 188
Side margin 200, 226, 279
Sideways scrolling increment 282
Sigma 94
SIMPLE 306
Simple letter example 346
Simple printer driver 104
Single sheets 110, 206, 318
SK 225, 254, 270
Skip printing 254
Skip word 127
SL 225
SM 226
Snaking 112
Soft character 369
Soft hyphen 86, 88, 97
Soft return 85
Soft spaces 85, 166
Sorting method 282
Spaces 85-88, 97-98, 166, 279
  hard 98
  non-break 86
  soft 85, 166
Spanish 94
SPELL 128, 188
Spell check whilst typing 130, 280
Spell checking 121
  block 128
  file 128
  text 124, 128
  word 128
Spell checking language 225, 280
Spell ignore marker 61, 88, 130
Spell memory 280
Spell menu 131
SPELLB 128, 188
Spelling checker options 280
SPLIT 189
Split screen editing 103
Splitting file 189
Splitting lines 62
Splitting variables 269
SPOFF 189
SPON 189
SPOOL 189
Spooling to file 189
SPOOLOFF 189
Spreadsheets 235
SS 306
ST 190, 226, 296
ST Character Set 362
Standard letter 234
Star LC24 119
Startup disc 19
STAT 190
STATUS 147, 190
Status line 54, 57, 72, 369
Sticky key 285
Sticky shift keys 49
STOP 190
Stop printing 226
Store word 126
Stored command 369
Stored commands 46, 50, 84, 109, 111, 183, 195, 204, 270
  overview 195
  printing 111
String 87, 230, 369
Subdirectory 146, 175
Submenu 43
Subscript 98, 105
Substrings 269
Sub-directories 68
Summary of Protext Commands 323
Superscript 98, 105
SV 226, 247, 266
SVK 187
SW 190
SWAP 190
Swapping characters 57
Swapping documents 190
Swedish 94
SYM 190
SYMBOL 190
Symbols 92
SYS 273-274
System variables 274

T 192
TAB 48, 76, 78, 191
Tab leader 197
Table of contents 136
Tables of figures 241
Tabs 76, 78, 81, 88, 98, 110, 166, 183, 191, 241, 279, 282, 369
  decimal 78, 241
  right justification 78
Tau 94
Taxan 223
Template 196, 234, 240
Template Files 196
Temporary files 147
Temporary text path 147, 280
Text 85, 97, 107, 166, 178, 185
  clearing 70
  counting 160
  fixing 166, 212
  formatting 166
  naming 176
  printing 107, 178
  saving 185
Text box 369

Text font 281
Text mode 53, 99
Tilde 92
TILDE 307
Time 96, 273
TIME 192
Timed automatic save 96
Title 85, 198
TL 226
TM 227
Toggle 369
Tokens 138-139, 159, 169, 369
Top margin 227, 279
TOUCH 192
Tree 146
TW 193
Type 66
TYPE 150, 192
Typesetting 113
TYPEWRITER 193
Typewriter mode 193
Typing file 192

Umlaut 92
UN 227, 259, 270
Undelete buffer 73
Undelete buffer size 280
Undeleting 57, 73
Underlining 98, 104-105
Until 259
UP 194
UPDATE 194
Update dictionary 280
Updates 42
Upgrade 42
Upper case 56, 369
Upr 231
Using a RAM disc 350
Utility 369

Variables 207, 229, 241-242, 263, 265, 270, 274
  asking 247
  joining 266
  setting 247
  splitting 269
Version numbers 42
Vertical scrolling 62
VGA 98
Viewing files 66
Viewing the text 97

WC 227, 261
WF 227-228, 261
WIDTH 309
WIDTHS 310
WIDTHS.PP 315
Wildcards 88, 90, 150, 155, 165, 369
WIN 194
Window 152, 287
WINDOW 194
Window resizing 100
WM 228
Word 87, 126-127, 369
  deleting 56
  storing 126
Word pattern 155
Word processing 13
Word processor 369
Word wrap 83, 279
Words
  counting 70, 160
WordStar 281, 297, 339
Workbench 35, 153, 369
Wrapping in quotes 238-239
Write file 261
Writing to file 227, 260
WT 228
Wysiwyg 98

X 163
XS 164
XT 14, 169

Year 273

ZM 228

0 194

1 194

2 194

43 line display 99

50 line display 99